

General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.
- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.
- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.
- This document is paginated as submitted by the original source.
- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

MARCH 1976

NASA CR- 144 770

MDC G6210

(NASA-CR-144770) A METHODOLOGY FOR
PRODUCING RELIABLE SOFTWARE, VOLUME 2 Final
Report (McDonnell-Douglas Astronautics Co.)
339 p HC \$10.00

CSCL 09B

N76-29946

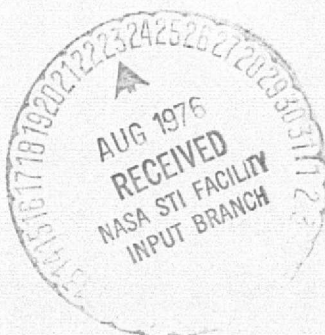
Unclas

G3/61 48522

**FINAL REPORT
A METHODOLOGY FOR PRODUCING
RELIABLE SOFTWARE**

VOLUME II

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY



MCDONNELL DOUGLAS



CORPORATION

**MCDONNELL
DOUGLAS**



**FINAL REPORT
A METHODOLOGY FOR PRODUCING
RELIABLE SOFTWARE**

VOLUME II

MARCH 1976

MDC G6210

PRINCIPAL INVESTIGATOR:

Leon G. Stucki

LEON G. STUCKI

PROJECT MANAGER

AUTOMATED VERIFICATION SYSTEMS

OTHER CONTRIBUTORS:

PAUL MORANDA

GARY FOSHEE

MARJORIE KIRCHOFF

ROGER OMRE

APPROVED BY:

Zygmunt Jelinski

ZYGMUNT JELINSKI

MANAGER

COMPUTER SCIENCES

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY-WEST

5301 Bolsa Avenue, Huntington Beach, CA 92647

PREFACE

This report was prepared for Mr. Evmenios Damon, Mission Operations Computing Division, NASA Goddard Space Flight Center under Contract Number NAS 5-20781.

Requests for further information or assistance will be welcomed by McDonnell Douglas representatives:

- . L. G. Stucki, Study Manager
Huntington Beach, California
Telephone: 714-896-3774
- . M. W. Hogan, Contract Administrator
Huntington Beach, California
Telephone: 714-896-4856

APPENDIX A
AUTOMATED VERIFICATION TOOLS

CONTENTS

Section A.1	INTRODUCTION
Section A.2	DESCRIPTIONS OF THE AUTOMATED TOOLS TESTED
Section A.3	TIMING STATISTICS OBTAINED FROM THE TEST RUNS
Section A.4	COMPUTER GENERATED OUTPUT FROM THE TEST RUNS
Section A.5	SUMMARY OF TEST RESULTS
Section A.6	RECOMMENDATIONS FOR THE USE OF AUTOMATED VERIFICATION TOOLS BY NASA

A.1 INTRODUCTION

The purpose of this task is to investigate the existence, availability, and applicability of automated verification tools. A comparative analysis is made and relative merits are evaluated. In addition, recommendations regarding development of new tools or modification of existing tools for NASA applications are discussed.

In the course of this investigation eight automated verification tools were tested. Two difficulties were present from the outset. The first was the availability of certain tools for purposes of a comparative analysis. Researchers were reluctant to release their tools for an evaluation over which they had no control. Thus some better known tools such as PACE and RXVP were not included in the study. Secondly, due to the diversity of tool functions, capabilities and machine implementations, a suitable test bed for comparison had to be found. This was accomplished by using the same source program for testing the tools and adopting common units of measurement for timing statistics. These units of measurement are:

- (1) the time required to compile the test program
- (2) the time required to execute the test program

Thus, if a particular tool was tested on machine X, the timing statistics are relative to the normal compile and execution times of the program on machine X.

A.2 DESCRIPTIONS OF THE AUTOMATED TOOLS TESTED

A.2.1 Automatic Test Data Generator (ATDG)

TRW
Houston, Texas

Implementations: UNIVAC

ATDG examines a FORTRAN source program to determine the values of program variables required to produce optimal test cases. A report is produced upon completion of a path analysis for each program or subroutine. The report is presented in six sections:

- Section I: FORTRAN source code listing and corresponding segment numbers. (A segment is a set of contiguous source lines with no branch points).
- Section II: Static error analysis. Checks are made for infinite loops, unreachable code, and set/use violations.
- Section III: Segment number reference for symbols in set/use statements.
- Section IV: List of program paths
- Section V: List of program paths with branch expression values which affect execution.
- Section VI: Skeleton listing of each program path.

Exhibit Reference: A

Test results:

The exhibit shows the ATDG reports produced for FORTRAN program HSMAN and one subroutine.

The output from the ATDG static error analysis (sections II and III) is interesting only when errors are uncovered in the element being analyzed. Of the six elements, only the main program (HSMAN) had any detectable errors. The only errors which may require some explanation are those related to unreachable code:

- a) ATDG says that the transfer 67-69 (and consequently 69-70 and 70-50) cannot be reached. An attempt was made to verify this. The transfer 67-69 is executed if $L-N+1=0$ (i.e., if $L=N+1$); since $L=N$ is set at segment 59 and L is decremented at segment 62, it appears that L can never be greater than N . (The reader is encouraged to examine the code and locate these transfers).
- b) ATDG says that the transfer 96-97 (and consequently 97-113) cannot be reached. Since $NVEC=4$ is set at segment 1 and is not altered until segment 105, it appears that $NVEC.EQ.0$ will in fact always be false at segment 96.

The final data generation output of ATDG is most useful if the transfers in the element are dependent upon integer switch variables, especially if these are inputs to the element or are set to a constant. It does very badly if switching is based upon dimensioned variables, since it does not interpret the value of the dimensions. When processing DO loops, it iterates the loop only as many times as are necessary to execute all transfers within the range of DO (i.e., the indexes on the DO statement are disregarded). Since the elements in this program perform much of the transfers based upon the value of A(I,J) and all contain a large number of DO loops, much of the data generation information is difficult to interpret. To assist the reader in understanding the reports, the following test cases have been derived for subroutine COMPIT:

- a) By looking at Section VI, we know that the inputs which affect branching are N, L, SIGMA, A, and RHO. These are the values which will constitute a test case; all other inputs are arbitrary.
- b) The listing (section I) shows that the key feature of the routine is the loop from segment 2 to 36. The path 1 in Section V shows that the loop (i.e., segment 36) is exercised 3 times, therefore $M-L=2$. We can thus assume that $L=1$ and $N=4$ will be alright since, in segment 1, $M=N-1=3$ would then be set.
- c) It appears that, after deriving a setting for L and N, the only other constraints to set are those on B1 and D (i.e., since these are the branches shown in output V that are not dependent upon L and N). Section V shows that the first time through the loop, the conditions B1.GE.0 and D.NE.0 must be true; the second loop, B1.LT.0 and D.NE.0 must be true; the third loop B1.GE.0 and D.NE.0 must be true. Further, D cannot be negative, so change all D.NE.0 to D.GT.0. Then, the conditions for Path 1 can be satisfied by:

First Loop

$$D = \text{SQRT}(B1^2 + B2^2 + B3^2) > 0 \text{ true if } B1 \neq 0$$

$$B1 = A(1,1)(A(1,1)-\text{SIGMA})+A(1,2)A(2,1)+\text{RHO} > 0$$

$$\text{true if } \underline{A(1,1)=1.0}, \underline{\text{SIGMA}=.5}, \underline{A(1,2)=A(2,1)=\text{RHO}=0.0}$$

Second Loop

$$D > 0 \text{ true if } B1 \neq 0$$

$$B1 = A(2,2)(A(2,2)-\text{SIGMA})+A(2,3)A(3,2)+\text{RHO} < 0$$

$$\text{true if } \underline{A(2,2)=.1}, \underline{\text{SIGMA}=.5}, \underline{A(2,3)=A(3,2)=\text{RHO}=0.0}$$

Third Loop

D>0 true if B1≠0

B1=A(3,3)(A(3,3)-SIGMA)+A(3,4)A(4,3)+RHO·0

true if A(3,3)=1.0, SIGMA=.5, A(3,4)=A(4,3)=RHO=0.0

Then inputs for Path 2 can easily be derived as:

N=2, A(1,1)=A(1,2)=A(2,1)=RHO=0.0, SIGMA=.5

A.2.2 FORTUNE

CAPEX
Phoenix, Arizona

Implementations: IBM, CDC

FORTUNE is used for FORTRAN source level optimization and execution analysis. Statistics gathered during program execution provide:

- (1) the exact number of times each statement was executed,
- (2) the estimated total execution time for each statement,
- (3) the number of IF statement true paths taken,
- (4) a summary of the estimated time and percentage of time spent within each program module.

Standard preprocessor/postprocessor organization is used. Timing information is gathered by analyzing each source statement, estimating the execution time for each statement, and then multiplying the estimated time by the number of executions. Estimates are obtained as follows:

- (1) a basic cost for each statement type (DO, GOTO, etc.),
- (2) a further incremental cost for each operation in arithmetic statements, each argument in subroutine entries, and invocation of intrinsic functions.

Estimates are relative to a base value of 2 for a GOTO statement. Library subroutine times are based on published values. Timings are not estimated for I/O statements.

Exhibit Reference: B

Test results:

The exhibit shows the results of instrumenting an entire FORTRAN program. Note the use of an annotated source listing for reporting data.

A.2.3 NBS Analyzer

Gordon Lyon
Rona Stillman
Institute of Computer Sciences and Technology
National Bureau of Standards
Washington, D.C. 20234

Implementations: UNIVAC

The NBS Analyzer is a simple static and dynamic tool for use with FORTRAN programs. Static analysis consists of a summary of source statements by FORTRAN type (i.e., GOTO, DO, FORMAT, etc.). To obtain dynamic statistics, a preprocessor divides the source program into segments. A segment is defined as a set of contiguous instructions containing no transfer of control statements. Thus a new segment is defined upon encountering a GOTO, IF, DO, etc. statement. Assignment of segments continues across subroutine boundaries; that is, a main program and any attendant subroutines are treated as a single program. Dynamic statistics consist of a matrix containing execution counts for each segment of the instrumented FORTRAN program.

Exhibit Reference: C

Test results

Exhibit pages 1C through 15C show the segment assignments for a FORTRAN program with five subroutines. Page 16C contains the Local Analysis for the program. Dynamic execution counts are contained on page 17C.

A.2.4 Program Evaluator and Tester (PET)

McDonnell Douglas Automation Company
Huntington Beach, California

Implementations: IBM, CDC, GE,
Honeywell

PET is an automated aid used in debugging, testing and documentation of FORTRAN programs. Standard preprocessor/postprocessor organization is used. Run time statistics include:

- (1) the number of times each statement was executed
- (2) the number of times each branch of a program was executed. This includes branch counts for logical and arithmetic IF conditions, plus computed and assigned GOTO's branching histories.
- (3) the minimum and maximum values attained by an assignment statement variable or DO loop parameter.
- (4) the first and last values attained by an assignment statement variable or DO loop parameter.
- (5) relative subroutine execution timings.

All of the above except (1) and (2) are optional, and may be activated via an option card. For each subprogram, a one page syntactic and operational profile is written by the postprocessor. For each program, four one page reports are written: a syntactic and operational profile, a subprogram operational summary, a subprogram execution summary and a subprogram timing summary.

Exhibit Reference: D,E

Test results:

The exhibit shows the results of instrumenting an entire FORTRAN program. Exhibit D reflects a run with full instrumentation, while Exhibit E shows the same program executed with minimal instrumentation. Note the expanded use of the annotated source listing for displaying dynamic execution statistics.

A.2.5 PFORT Verifier

A. D. Hall
B. G. Ryder
Bell Laboratories
Murray Hill, New Jersey

Implementations: IBM, CDC,
Honeywell,
Burroughs,
Others

The PFORT Verifier checks a FORTRAN source program for adherence to PFORT, a portable subset of ANS FORTRAN. Subprogram communication is checked through common argument lists. Debugging and documentation aids include subprogram cross reference giving type, usage and attributes of each identifier with a list of statements in which it occurs. Also provided is a summary by subprogram listing argument attributes, common blocks used, subprograms called, and the calling subprograms. PFORT is written in PFORT.

Exhibit Reference: F

Test results:

Exhibit pages 1F through 6F illustrate the results obtained after scanning subprogram IMVEC. Exhibit page 7F is the summary report for the entire program.

A.2.6 Problem Program Evaluator (PPE)

Boole and Babbage
Sunnyvale, California

Implementations: IBM

PPE is a performance measurement tool residing in the same region as a problem program to be measured. The executing program is sampled at regular intervals by the PPE Extractor and the sample data is stored on a disk or tape for later analysis by the PPE Analyzer. Measurements include:

- (1) percent of time the program is actively executing,
- (2) percent of time the program is waiting for I/O to complete,
- (3) percent of time the program is waiting for non-I/O events to complete,
- (4) how the I/O is distributed among the problem program's data sets,
- (5) percent of time the problem program is executing SVC's (Supervisor Calls),
- (6) where in the program the various activities and waits occurred.

Note that PEE performs analysis on load modules, and is therefore independent of the programming language used.

Exhibit Reference: G

Test results:

The exhibit displays the results of a PPE extractor and analyzer run for a COBOL program. Since the PPE Code Activity Report is rather complicated, the significant portions of the report are explained below:

1. Distribution of Activity (Excluding DSOW Wait)

The "Sample Boundary" referred to is the entire region specified. Samples taken, above and below the sample boundaries, are those in OS routines which cannot be related back to the problem program. "Active" is defined as that percentage of activity when instructions are being executed within the area. "Wait" is that activity when the CPU must wait for some event associated with an activity within the area before it can resume processing -- not including I/O wait that can be associated with a data set; examples are OPEN, CLOSE, LINK and LOAD.

2. Distribution of DSOW Wait

This section gives a comparison, on a percentage basis, of the number of DSOW samples associated with each data set for which there is a DD statement. Data sets showing a high percentage of activity should be examined for opportunities to reduce this amount.

3. Module Map (Excluding Wait)

This section provides a list of each module sampled by the extractor in the order in which they were encountered; the starting and ending addresses for each module, relative to the beginning of the OS region; and the percentage of active samples for each module.

4. Study Report

This section is the most important part of the report. It provides information as to how the CPU time was distributed throughout the program. The "Code Execution Frequency" sub-report lists the percentage of time "Active" samples were taken in each interval of the program. An interval is a number of contiguous bytes (usually 32) used for grouping extractor results. Therefore, each interval shown on the report represents several machine instructions. The starting and ending location shown for each interval represent their addresses relative to the beginning of the module.

For convenience, the 10 Most Frequently Executed Intervals are listed separately. The value of this report is that it allows the Programmer to concentrate his effort of optimizing the source code on those areas of the program where the most time is being consumed. For those intervals having a high percentage, the corresponding source statements should be examined, in an attempt to improve the coding.

A drawback of PPE is the necessity of consulting a source program load map to associate timing statistics. An alternate approach is taken by FORTUNE, in which timing values are reported directly on the source program listing. PET also provides similar statistics on a more summarized basis per subprogram. An advantage over most other performance measurement tools is PPE's ability to measure programs of any source language on IBM.

A.2.7 FORTTRAN Structuring Engine (SFORTTRAN)

Caine, Farber and Gordon, Inc.
Pasadena, California

Implementations: IBM

The FORTRAN Structuring Engine produces a SFORTTRAN source program and listing from a FORTRAN source program. SFORTTRAN is a highly structured superset of FORTRAN. A SFORTTRAN to FORTRAN preprocessor is available to allow compilation and execution of SFORTTRAN programs. The SFORTTRAN program listing is designed to easily identify blocks of code and thus obtain a picture of the overall logical structure of the program.

Exhibit Reference: H

Test results:

Probably the best produce of this type on the market today, in terms of structuring a given FORTRAN program. Note that even if the preprocessor is never used to execute the SFORTTRAN representation of a particular FORTRAN program, the SFORTTRAN listing can be an invaluable aid in determining what is going on in a large and complicated program.

The Structuring Engine has serious practical limitations to go along with its sophisticated output, however, since a 1000K load module size makes it rather machine dependent. Also, the number of SFORTTRAN lines starts to go up extremely fast as the number of branch points in the FORTRAN program increases.

A.2.8 DISSECT Symbolic Evaluation System

McDonnell Douglas Automation Company
Huntington Beach, California

Implementations: PDP-10

DISSECT is a symbolic evaluation system that is used to analyze programs written in ANSI FORTRAN.

When a program path is executed by running the program on a given input, the correctness of the path for that input can be determined by examining the effects of the calculations carried out by the path. If the path is executed "symbolically" rather than with actual data, it may be possible to use a single execution to illustrate its correctness on a large subset of the input domain rather than on just a single value. Symbolic execution of a program is carried out by given dummy symbolic values rather than actual numeric (string, logical, etc.) values to all or some of the input variables of the program. An expression in the program involving variables with symbolic values is evaluated by substituting the current symbolic values of the variables into the expression. The resulting expression is then simplified algebraically. All operators having only actual as opposed to symbolic operands are evaluated in the normal way. The resulting expression is the symbolic value of the original expression.

The DISSECT system operates on two input files and produces an output file. One of the input files contains a FORTRAN program to be analyzed and the other contains the DISSECT commands which tell the system what kind of symbolic evaluation analysis to carry out. The output file contains the results of the analysis.

The command file for a DISSECT analysis is divided into a number of cases. The program in the input file is processed completely for each case. Each case has a section for a text description of the part of the program to be analyzed by the case. This text is not processed by the system and is considered to be the specification for the case. The rest of a case contains commands which identify the part of the program which corresponds to the case, commands which assign (symbolic) values to variables and commands which specify what output is to be generated.

The output file which is generated by a DISSECT analysis is also divided into cases. Each output case contains the case specifications and commands as well as the output generated by the system for that case. The user can check the validity of his program by comparing case specifications with system output for cases.

A program path is a possible flow of control through the program. A path is feasible if at least one element of the program's input domain causes that path to be executed. In general, a complete set of DISSECT cases for a program should "cover" the program in some sense. One approach is to analyze each feasible path (up to some number of iterations of loops). Complex programs having many paths can be divided into segments and analyzed using separate cases.

Exhibit Reference: I

Test Results:

The first page of the exhibit contains a program called INTERP, where DISSECT was used to verify that the program agreed with its specifications. The INTERP routine is written so that the number of points NTERMS used in the interpolation process may be less than the number of points available (NPTS). The first segment of the routine, lines 1-25, decides which points to use. It involves choosing a value for I1. The points X(I1), X(I1+1), ..., X(I1+n-1) in the program correspond to the points x1, x2, ..., xn in the documented formulae (n=min {NPTS, NTERMS}). The documentation states that x1 (i.e., X(I1)) "is chosen so that x1 and xn straddle x". "If the value of x is too close to the lower or upper limit of the values of x1, the corresponding value of x1 or xn is set equal to the limiting value".

It was decided to examine the paths through this section of code for NPTS = 1 and 2 (the limiting cases) and also for NPTS = 3 and NPTS = 4. Exhibit pages 2I-33I contain the system output for these four cases, which cause DISSECT to analyze all paths in the program up to statement 28 for NPTS = 1, 2, 3 and 4.

This is a large number of paths but it was found that the output for each was easy to read and that it was easy to determine if the program was correct for the case.

The second segment of DISSECT, lines 26-33 is supposed to compute the following values of Δ and Δ_i .

$$\Delta = \frac{x - x_1}{x_2 - x_1} \quad \Delta_i = \frac{x_i - x_1}{x_2 - x_1}$$

The verification of this segment is contained in Cases 5 and 6 on exhibit pages 33I and 34I.

The Case 6 command ASSIGN I1 = "I1" ensures that the output for Case 6 will use the symbol I1 to stand for I1 rather than the value it may have as the result of earlier calculations.

Symbolic output for code in the third segment, lines 34 through 48, appears in Cases 7 and 8 on exhibit pages 35I and 36I.

The final segment of the program, lines 49 through 60, is supposed to compute the formula:

$$Y(x) = a_1 + \sum_{j=2}^n \{a_j \prod_{i=1}^{j-1} (-\Delta_i)\}.$$

Symbolic output for this segment appears in Cases 9 and 10 on exhibit pages 36I and 37I (the program uses the variable YOUT to represent the value y(x)).

A.2.9 JOYCE

McDonnell Douglas Automation Company
Huntington Beach, California

Implementations: CDC

JOYCE is a program designed to aid the programmer both in checkout and documentation. It accepts as primary input FORTRAN source decks in the form of card decks or CDC compile files. The source decks are edited producing tables of referenced symbols (variables, subroutines, and I/O files) and flowlists (microfilm FORTRAN listings with all transfers indicated by arrows to the right of the listing and all DO loops indicated by brackets on the left). Also available are cross-reference lists for variables, subroutines, and I/O references. The above edit information may be combined with definitions and mathematical symbols, input on data cards, to yield a completely cross-referenced program glossary. This glossary may be listed on microfilm in a variety of formats on a program and/or subroutine level sorted on FORTRAN and/or mathematical symbol or in any of several special modes (e.g. by storage location). All operating modes require only a few easily punched free-format control cards. JOYCE is written entirely in FORTRAN IV.

Exhibit Reference: J

Test results:

Exhibit pages 1J through 10J show the flowlist for the test program and its subroutines, produced on microfilm. All transfers are indicated by arrows to the right of the statement text, and all DO loops are indicated by brackets to the left.

Exhibit pages 11J through 24J contain the symbol cross references in alphabetical order. A description is given for each use of a symbol name, together with an analysis of the symbol's use in a subroutine. This output may optionally appear on a printer.

Entries under the Subroutine Usage heading include:

- (1) Subroutine name in which the symbol is used
- (2) A code indicating how the symbol is used. A complete list of usage codes is contained on Exhibit page 44J. The most frequent codes appearing in this example are:

- M - the symbol is modified
- W - working variable
- I - input variable
- O - output variable

- (3) Symbol Type (real, integer, etc.)
- (4) Variable name within the subroutine
- (5) Dimension, if symbol is an array

The Cross Reference Matrix is on Exhibit pages 44J through 46J. This matrix is a summary of the information in the cross reference discussed earlier. This output appeared on the system printer.

A.2.10 DAVE

Lloyd Fosdick and Leon Osterweil
Dept. of Computer Science
University of Colorado
Boulder, Colorado

Implementations: CDC, IBM

DAVE is a system used to detect the existence of a wide variety of errors in ANSI Standard FORTRAN programs. Common programming errors that DAVE will detect include:

- (1) misspelled variables
- (2) transposed variables
- (3) failure to initialize
- (4) misspelled labels

Under mild restrictions, DAVE can guarantee that certain types of errors are absent.

DAVE does not attempt to positively identify the exact nature of every error in a program. Instead, the program is probed for suspicious and elusive constructs, thus confronting the programmer directly with various subtleties and suspicious aspects of the program. The programmer must then use the messages and warnings produced by DAVE to improve his program.

DAVE is written in highly portable FORTRAN.

Exhibit Reference: K

Test results:

Exhibit pages 1K through 13K show the program listing for test program UNSEIG and its five subroutines. Page 14K illustrates the Call Graph Table, which lists the external calls for the main program and each subroutine.

The remainder of the Exhibit shows various warning messages for the main program and each subroutine.

REFERENCES

NBS Analyzer

1. G. Lyon and R. B. Stillman. A FORTRAN Analyzer. NBS Technical Note 849, Institute for Computer Sciences and Technology, National Bureau of Standards, Washington, D.C., October 1974.

PET

1. L. G. Stucki. A Prototype Automatic Testing Tool. Paper Number MDAC WD 1985, 1972.
2. L. G. Stucki. Automatic Generation of Self-Metric Software. Paper No. MDAC WD 2144, 1973.
3. C. B. Boettcher. Program Evaluator and Tester, CDC Users Manual. McDonnell Douglas Automation Company, Document Number M2085074, 1974.

PFORT

1. B. G. Ryder. The PFORT Verifier. Software Practice and Experience. Volume 4, Number 4, October - December 1974, pp. 359-378.
2. B. G. Ryder. The PFORT Verifier - User's Guide. Computing Science Technical Report #12, Bell Laboratories, Murray Hill, N.J.
3. A. D. Hall and B. G. Ryder. The PFORT Verifier - Installation and Maintenance. Bell Laboratories, Murray Hill, New Jersey, unpublished.

DISSECT

1. L. G. Stucki and W. E. Howden. Final Report - A Methodology for Effective Test Case Selection, Phase III. McDonnell Douglas paper number G6211, March 1976.

DAVE

1. L. D. Fosdick and L. J. Osterweil. DAVE - A FORTRAN Program Analysis System. Computer Science and Statistics: 8th Annual Symposium on the Interface, Los Angeles, February 1975.

A.3 TIMING STATISTICS

The following pages contain timing statistics for each of the tools tested to date. Two common units of measurement have been established to give some basis of comparison for all tools tested. These units of measurement are :

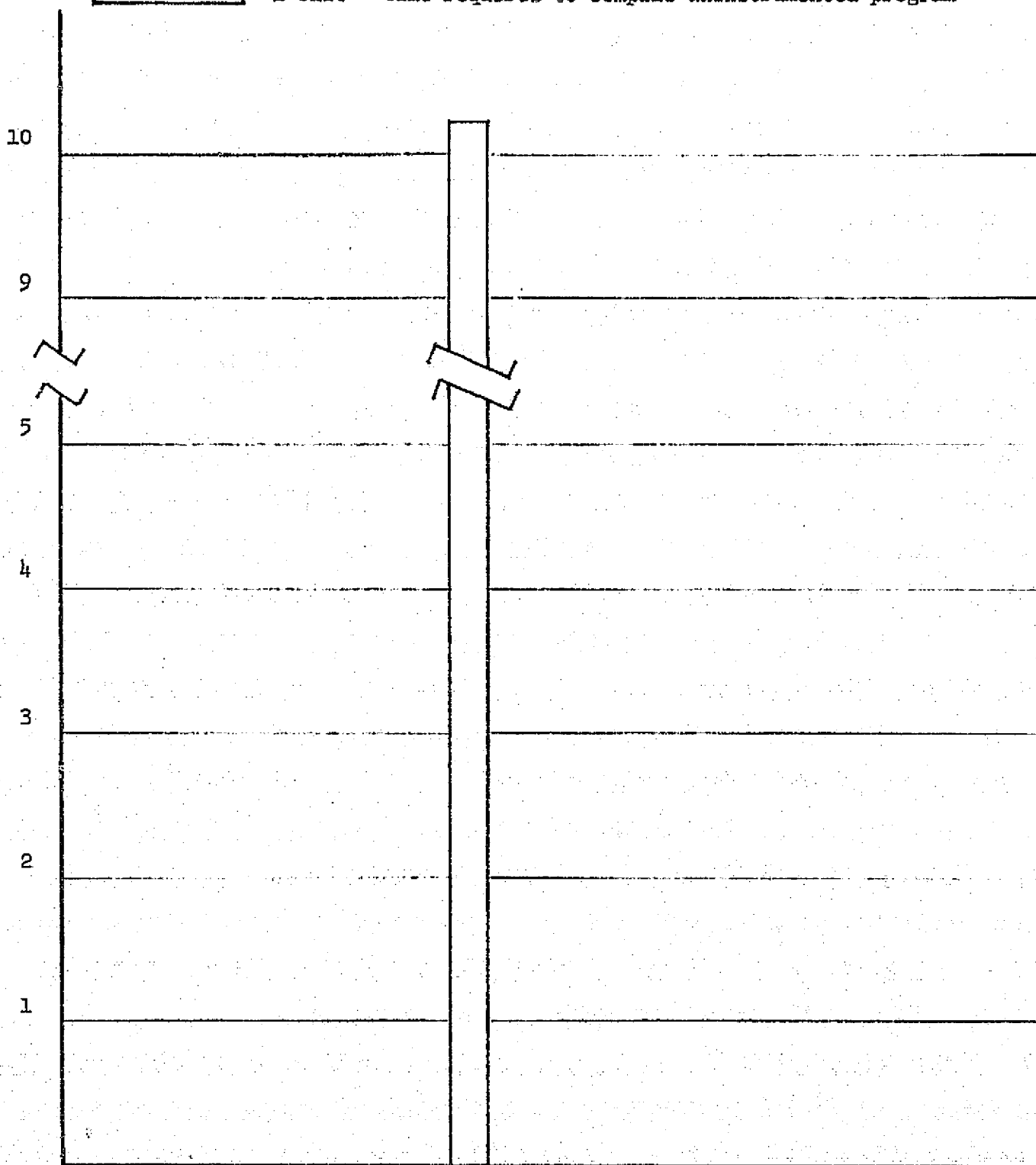
- (1) the time required to compile the uninstrumented source program
- (2) the time required to execute the uninstrumented program.

Thus, if a particular tool was tested on machine X, the timing statistics are relative to the normal compile and execution times of the program on machine X.

In addition, the same test program is used for different tools whenever possible.

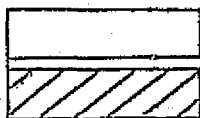


1 Unit = time required to compile uninstrumented program



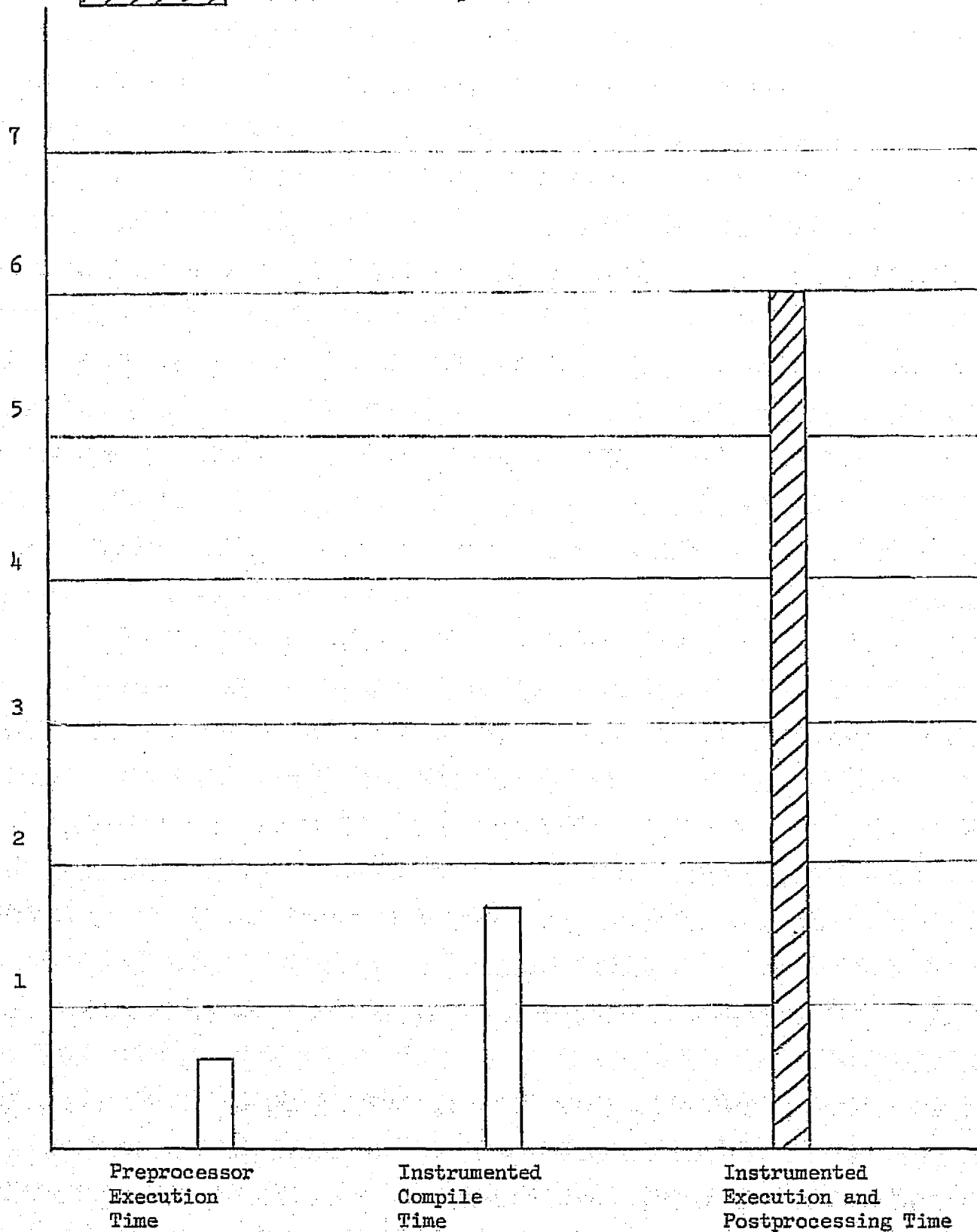
Execution Time

ATDG



1 Unit = Time required to compile uninstrumented prog

1 unit = time required to execute uninstrumented prog



FORTUNE



1 Unit = time required to compile and execute uninstrumented prog.

7

6

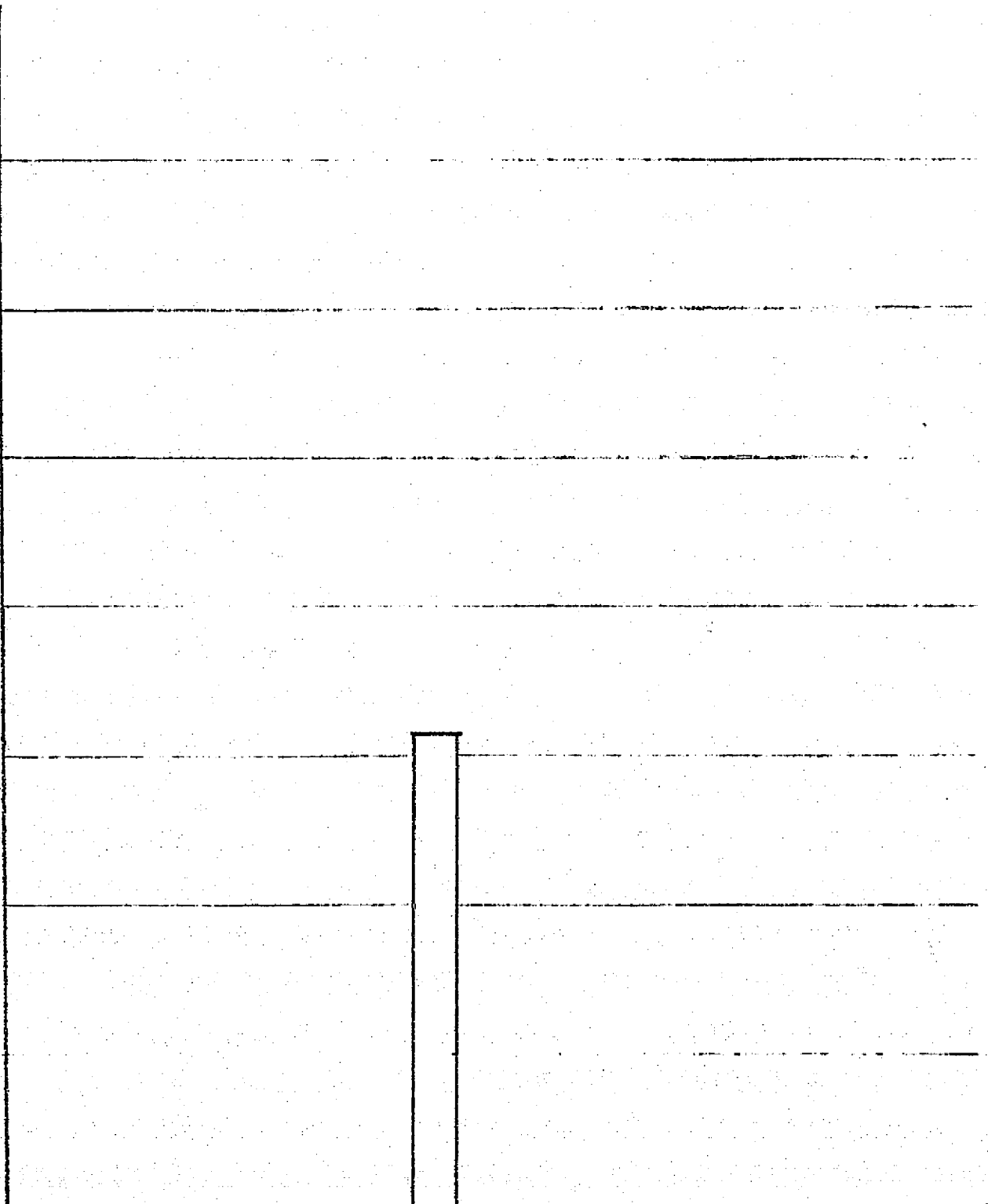
5

4

3

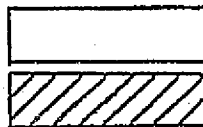
2

1



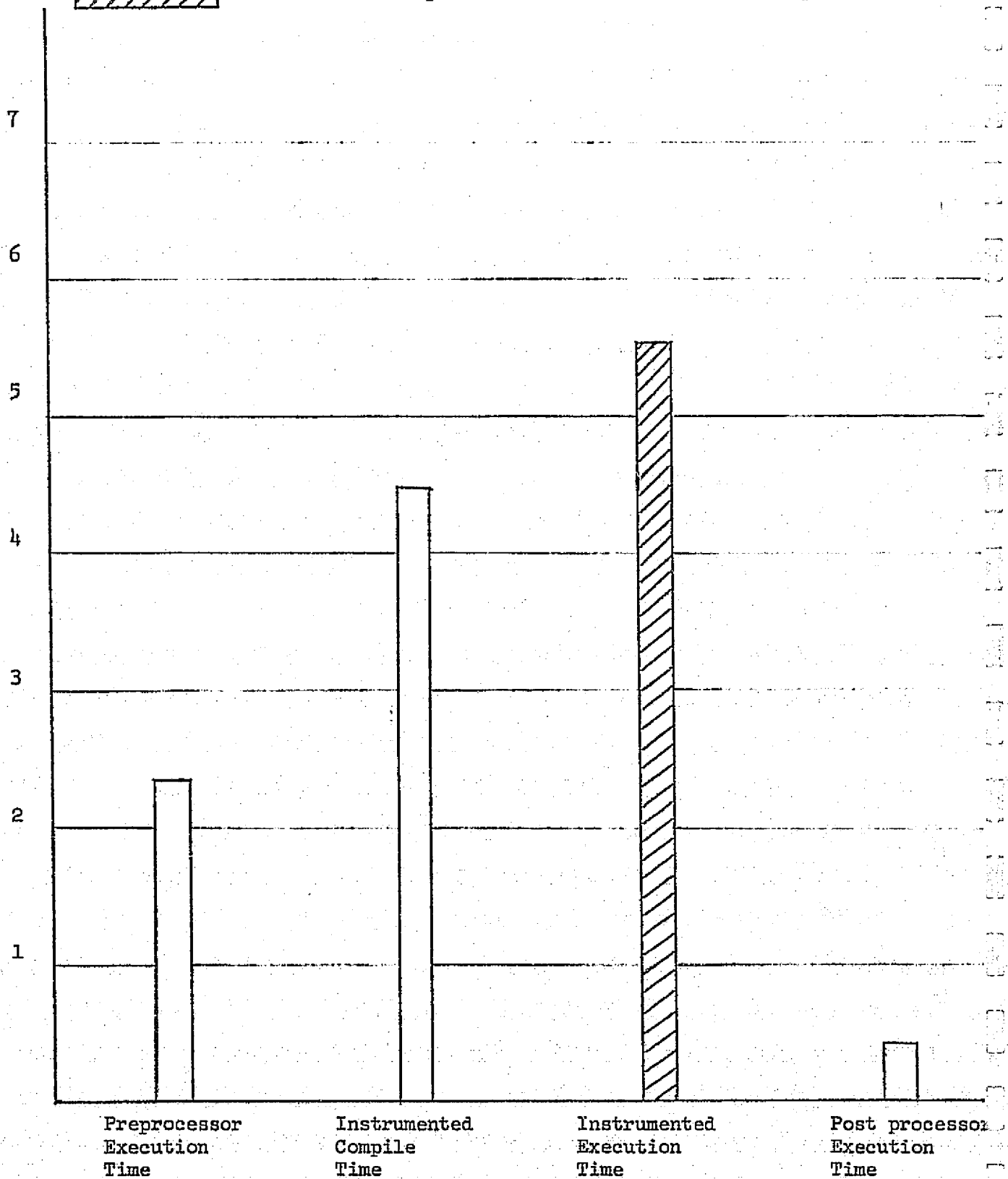
Total Execution
Time

NBS ANALYZER

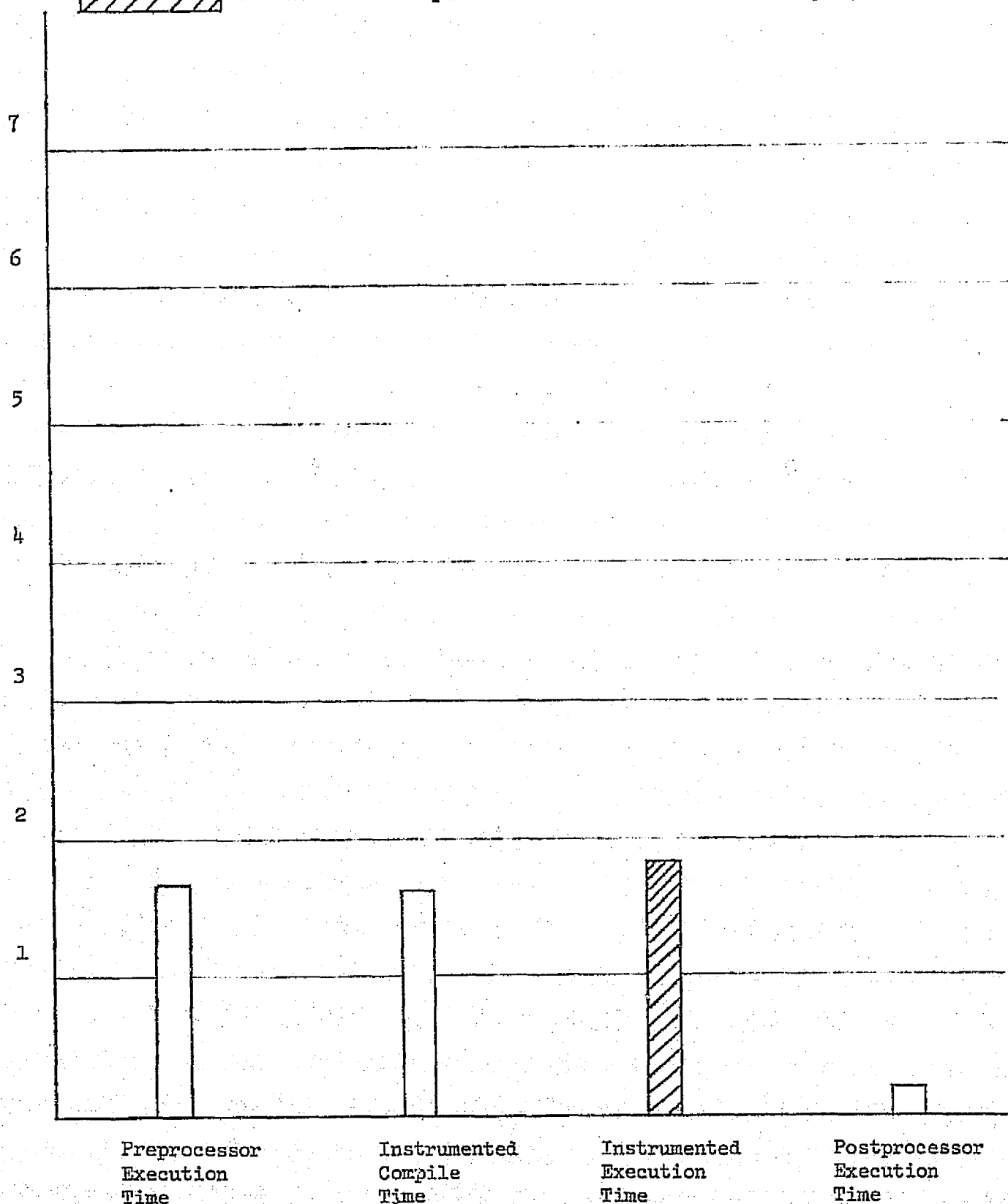
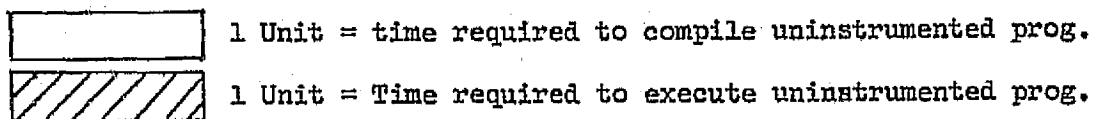


1 Unit = time required to compile uninstrumented prog.

1 Unit = time required to execute uninstrumented prog.



PET
(Full Instrumentation)



PET
(Minimal Instrumentation)



1 Unit = Time required to compile uninstrumented prog.

7

6

5

4

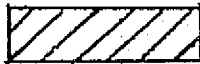
3

2

1

Execution Time

PFORT



1 Unit = Time required to execute uninstrumented prog.

7

6

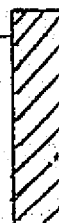
5

4

3

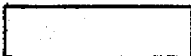
2

1



Extractor Execution Time

PPE



1 Unit = Time required to compile uninstrumented prog.

7
6
5
4
3
2
1

Execution Time

SFORTRAN



1 Unit = time required to compile uninstrumented prog.

7

6

5

4

3

2

1

Execution Time

DISSECT



1 Unit = time required to compile uninstrumented prog.

7

6

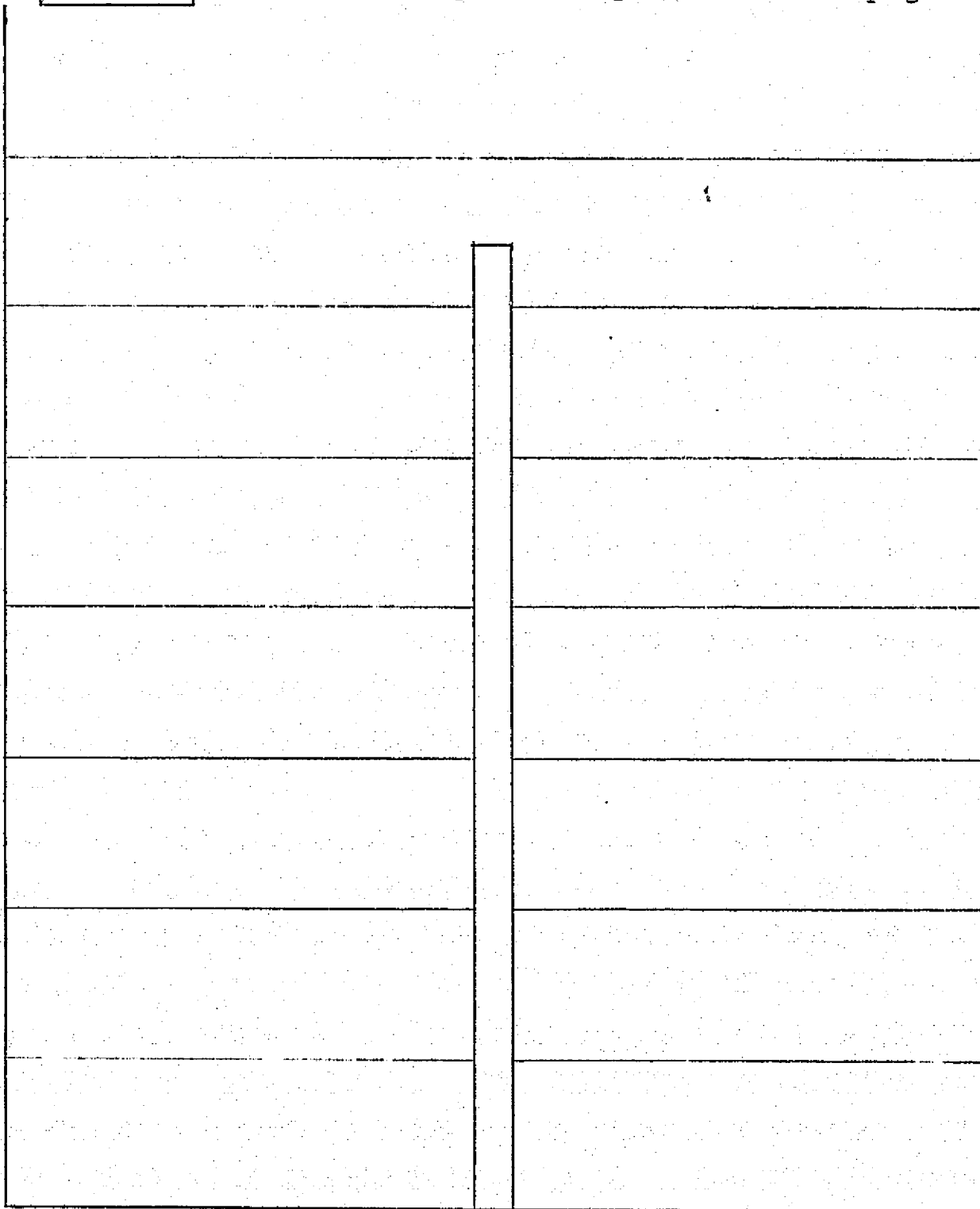
5

4

3

2

1



Execution Time

JOYCE

1 Unit = time required to compile uninstrumented program

16

15

5

4

3

2

1

Execution Time

DAVE

A.4 COMPUTER GENERATED OUTPUT FROM THE TEST RUNS

- A. Automatic Test Data Generator (ATDG)
- B. FORTUNE
- C. NBS Analyzer
- D. Program Evaluator and Tester (PET) - full instrumentation
- E. Program Evaluator and Tester (PET) - minimal instrumentation
- F. PFORT Verifier
- G. Problem Program Evaluator (PPE)
- H. FORTRAN Structuring Engine (SFORTAN)
- I. DISSECT Symbolic Evaluation System
- J. JOYCE
- K. DAVE

HSMAIN

PRECEDING PAGE BLANK NOT FILMED

1A

ID, U 14.

USE UPPER ASSUMED

14.01-06/02-16:54-(0,1)

III

SEGMENT NUMBERS	FORTTRAN STATEMENTS
--------------------	---------------------

0	ELEMENT HSPAIN
0	DIMENSION A(10,10),VEC(8), EIGVEC(4,4), INT
	*(10),CC(10),SS(10)
0	DIMENSION Z(10),D(10,10),BB(10,10)
1	NMAX = 10
1	NN = 4
1	NVEC = 4
1	N=NN
1	K=N-2
2	DO1J=1,K
2	K1=J+1
2	K2=J+2
2	GREAT= ABS(A(K1,J))
2	MAX=K1
3	DO2I=K2,N
3	AB= ABS(A(I,J))
3, 4	IF (GREAT.GE.AB) GO TO 2
5	3 GREAT=AB
5	MAX=I
6	2 CONTINUE
7	INT(J)=MAX
7, 8	IF (MAX.EQ.K1) GO TO 5
9	4 DO6L=J,N
9	E=A(K1,L)
9	A(K1,L)=A(MAX,L)
10	6 A(MAX,L)=E
11	DO7L=1,N
11	E=A(L,K1)
11	A(L,K1)=A(L,MAX)
12	7 A(L,MAX)=E
13	5 DO8I=K2,N
13, 14	IF (A(I,J).EQ.0.) GO TO 8
15	A(I,J)=-A(I,J)/A(K1,J)
16	DO11M=K1,N
17	11 A(I,M)=A(I,M)+A(I,J)*A(K1,M)
18	8 CONTINUE
19	DO 2000 M=K2,N
19, 20	IF (A(M,J).EQ.0.) GO TO 2000
21	10 DO 2001 I=1,N
22	2001 A(I,K1)=A(I,K1)-A(I,M)*A(M,J)
23	2000 CONTINUE
24	1 CONTINUE
25	DO 105 I=1,N
26	DO 105 J=1,N
27	105 D(I,J)=A(I,J)
28	DO 106 I=3,N

```

28      L=I-2
29      DO 106 J=1,L
30      106 A(I,J)=0.0
31      KI=N+1
31      R=0.0
31      C=0.0
32      DO 33 I=1,N
32      R=R+ ABS(A(I,I))
33      33 C=C+ ABS(A(I,N))
34      DO 34 I=2,N
34      RS=0.0
34      CS=0.0
34      L=I-1
35      DO 35 J=L,N
35      RS=RS+ ABS(A(I,J))
35      KIJ=KI-J
35      KII=KI-I
36      35 CS=CS+ ABS(A(KIJ,KII))
37, 38      IF (RS.LE.R) GO TO 36
39      37 R=RS
40, 41      36 IF (CS.LE.C) GO TO 34
42      44 C=CS
43      34 CONTINUE
44      IJK=I
44, 45      IF (R.GE.C) GO TO 54
46      53 DELTA=R*1.0E-11
46      DEL=R/17.1829
46      GO TO 153
47      54 DELTA=C*1.0E-11
47      DEL=C/17.1829
48      153 DO 107 I=1,N
49      107 A(I,I)=A(I,I)-DEL
50, 51      55 IF (N.NE.1) GO TO 48
52      47 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
53      GO TO 76
54, 55      48 IF (N.NE.2) GO TO 80
56      49 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
57      GO TO 76
58      80 AR1=0.0
58      AR3=0.0
58      AR4=0.0
59      61 L=N
60      DO 66 I=2,N
60      N2I=N+2-I
60, 61      IF (ABS(A(N2I,N2I-1)).LT.DELTA) GO TO 57
62      56 L=L-1
63, 64      57 IF (L.NE.N) GO TO 59
65      58 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
66      N=N-1
66      GO TO 48
67, 68      59 IF ((L-N+1).NE.0) GO TO 75
69      60 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
70      N=N-2
70      GO TO 55
71      75 B=A(N-1,N-1)+A(N,N)
71      C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
71      CI=A(N-1,N-1)-A(N,N)

```

ORIGINAL PAGE IS
OF POOR QUALITY

I

```

71          DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)
71, 72      IF (DISC.GE.0.) GO TO 63
73          62 R1=0.5*B
73          R2=.5* SQRT( ABS(DISC))
73          ANUM=(R1-AR3)*(R1-AR3)+(R2-AR4)*(R2-AR4)
73          DENOM=AR3*AR3+AR4*AR4
73, 74      IF (DENOM.NE.0.) GO TO 200
75          201 RHO=0.0
75          SIGMA=0.0
75          GO TO 203
76          200 RHO=C
76          SIGMA=B
77          203 AR3=R1
77          AR4=R2
77          CALL COMPIT(A,NMAX,L,N,RHO,SIGMA)
78          GO TO 61
79, 80      63 IF (B.GE.0.) GO TO 65
81          64 R1=.5*(B- SQRT(DISC))
81          GO TO 66
82          65 R1=.5*(B+ SQRT(DISC))
82, 83      IF (R1.NE.0.) GO TO 66
84          83 R2=0.0
84          GO TO 84
85          66 R2=C/R1
86, 87      84 IF( ABS(A(N,N)-R1).GE. ABS(A(N,N)-R2))GO TO
          *68
88          67 AR2=R1
88          GO TO 69
89          66 AR2=R2
90, 91      69 IF (AR1.NE.0.) GO TO 71
92          72 SHIFT=0.0
92          GO TO 73
93          71 SHIFT=AR2
94          73 AR1=AR2
94          CALL REALIT(A,NMAX,L,N,SHIFT,CC,SS)
95          GO TO 61
96, 97      76 IF (NVEC.EQ.0) GO TO 600
98          550 SMALL=DELTA*1.0E-30
98          NUM=0
98          JI=1
98          NM=NN+NN
99          513 C1=VEC(JI)
99          C2=VEC(JI+1)
99          NUM=NUM+1
100         DO 570 I=1,NN
101         DO 570 J=1,NN
102         570 A(I,J)=D(I,J)
103, 104     IF (C2.NE.0.) GO TO 561
105         CALL RVEC(A,NMAX,INT,C1,SMALL,NN,NUM,EIGVEC,
          *NVEC,CC,SS)
106         JI=JI+2
106, 107     IF (NVEC.EQ.NUM)GO TO 600
108         GO TO 513
109         561 CALL IMVEC(A,NMAX,INT,C1,C2,SMALL,NN,NUM,EIG
          *VEC,NVEC,CC,SS,2,BB)
110         JI=JI+4
110, 111     IF (NVEC.EQ.NUM)GO TO 600

```

I

```
112      GO TO 513
113      600 CONTINUE
113      WRITE(6,700) (VEC(I),I=1,NN)
114      DO 20 I=1,NN
115      20  WRITE(6,700) (EIGVEC(I,J),J=1,NN)
116      700  FORMAT(1H,5E20.8 / 5E20.8)
116      STOP
116      END
```

SEGMENT
NUMBERS

FORTRAN STATEMENTS

0

ELEMENT EOF

CAN:181

OF:181

INES:181 FIELDATA

BRKPT PRINTS

0,0 11.

USE UPPER ASSUMED

14.01-06/03-19:28-(0,1)

III

***CHECK SET/USE OF INTERNAL SYMBOLS

PROCESS SYMBOL A

NO SET VIOLATIONS HAVE BEEN DETECTED

A IS USED IN THE FOLLOWING SEGMENTS WITHOUT HAVING BEEN SET

	2	3	9	13	15	17	19	22	27			
SET SEGMENTS/	10	12	30	52	56	65	69	77	94			
	102	105	109									
USE SEGMENTS/	2	3	9	11	13	15	17	19	22			
	27	32	33	35	36	49	52	56	60			
	65	69	71	77	86	94	105	109				

PROCESS SYMBOL AB

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 3
USE SEGMENTS/ 3 5

PROCESS SYMBOL ANUM

ANUM IS SET IN THE FOLLOWING SEGMENTS BUT NEVER USED
73

PROCESS SYMBOL AR1

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 58 94
USE SEGMENTS/ 90

PROCESS SYMBOL AR2

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 88 89
USE SEGMENTS/ 93 94

PROCESS SYMBOL AR3

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 58 77
USE SEGMENTS/ 73

PROCESS SYMBOL AR4

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 58 77
USE SEGMENTS/ 73

PROCESS SYMBOL B

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 71

ORIGINAL PAGE IS
OF POOR QUALITY

III

USE SEGMENTS/ 75 76 79 81 82

~~PROCESS SYMBOL BB~~

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 109

USE SEGMENTS/ 109

~~PROCESS SYMBOL C~~

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 31 42 71

USE SEGMENTS/ 33 40 44 47 76 85

~~PROCESS SYMBOL CC~~

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 94 105 109

USE SEGMENTS/ 94 105 109

~~PROCESS SYMBOL CS~~

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 34

USE SEGMENTS/ 36 40 42

~~PROCESS SYMBOL C1~~

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 71 99 105 109

USE SEGMENTS/ 71 105 109

~~PROCESS SYMBOL C2~~

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 99 109

USE SEGMENTS/ 105 109

~~PROCESS SYMBOL D~~

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 27

USE SEGMENTS/ 102

~~PROCESS SYMBOL DEL~~

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 46 47 52 56 65 69

USE SEGMENTS/ 49 52 56 65 69

~~PROCESS SYMBOL DELTA~~

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 46 47

USE SEGMENTS/ 60 98

~~PROCESS SYMBOL DENOM~~

III

NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 73
USE SEGMENTS/ 73

P R O C E S S S Y M B O L D I S C
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 71
USE SEGMENTS/ 71 73 81 82

P R O C E S S S Y M B O L E
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 9 11
USE SEGMENTS/ 10 12

P R O C E S S S Y M B O L E I G V E C
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 105 109
USE SEGMENTS/ 105 109 115

P R O C E S S S Y M B O L G R E A T
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 2 5
USE SEGMENTS/ 3

P R O C E S S S Y M B O L I
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 3 13 21 25 28 32 34 48 63
100 114
USE SEGMENTS/ 3 5 13 15 17 22 27 28 33
32 33 34 35 49 63 102 113 115

P R O C E S S S Y M B O L I J K
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 44 52 56 65 69
USE SEGMENTS/ 52 56 65 69

P R O C E S S S Y M B O L I N T
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 7 105 109
USE SEGMENTS/ 105 109

P R O C E S S S Y M B O L J
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 2 25 29 35 101
USE SEGMENTS/ 2 3 7 9 13 15 17 19 22
27 37 35 102 115

ORIGINAL PAGE IS
OF POOR QUALITY

P R O C E S S S Y M B O L J I

SA

NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 98
USE SEGMENTS/ 99 106 110

PROCESS SYMBOL K
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 1
USE SEGMENTS/ 2

PROCESS SYMBOL K1
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 2 31
USE SEGMENTS/ 2 7 9 11 15 16 17 22 35

PROCESS SYMBOL KIT
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 35
USE SEGMENTS/ 36

PROCESS SYMBOL K1J
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 35
USE SEGMENTS/ 36

PROCESS SYMBOL K2
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 2
USE SEGMENTS/ 3 13 19

PROCESS SYMBOL L
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 9 11 28 34 59 77 94
USE SEGMENTS/ 9 10 11 12 29 35 62 63 67
77 94

PROCESS SYMBOL M
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 16 19
USE SEGMENTS/ 17 19 22

PROCESS SYMBOL MAX
NO SET VIOLATIONS HAVE BEEN DETECTED
NO JSE VIOLATIONS HAVE BEEN DETECTED
SET SEGMENTS/ 2 5
USE SEGMENTS/ 7 9 10 11 12

PROCESS SYMBOL N
NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/	1	52	56	65	69	77	94		
USE SEGMENTS/	1	3	9	11	13	16	19	21	25
		26	28	31	32	33	34	35	48
		52	54	56	59	60	63	65	66
		69	70	71	77	86	94		

PROCESS SYMBOL NM

NM IS SET IN THE FOLLOWING SEGMENTS BUT NEVER USED
98

PROCESS SYMBOL NMAX

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 1 52 56 65 69 77 94 105 109

USE SEGMENTS/ 52 56 65 69 77 94 105 109

PROCESS SYMBOL MN

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 1 105 109

USE SEGMENTS/ 1 98 100 101 105 109 113 114 115

PROCESS SYMBOL NM

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 98 105 109

USE SEGMENTS/ 99 105 106 109 110

PROCESS SYMBOL NVEC

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 1 105 109

USE SEGMENTS/ 96 105 106 109 110

PROCESS SYMBOL N2I

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 60

USE SEGMENTS/ 60

PROCESS SYMBOL R

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 31 39

USE SEGMENTS/ 32 37 44 46

PROCESS SYMBOL R40

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 75 75 77

USE SEGMENTS/ 77

PROCESS SYMBOL RS

NO SET VIOLATIONS HAVE BEEN DETECTED

ORIGINAL PAGE IS
OF POOR QUALITY

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 34

USE SEGMENTS/ 35 37 39

P R O C E S S S Y M B O L R1

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 73 81 82

USE SEGMENTS/ 73 77 82 85 86 88

P R O C E S S S Y M B O L R2

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 73 84 85

USE SEGMENTS/ 73 77 86 89

P R O C E S S S Y M B O L SHFT

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 92 93 94

USE SEGMENTS/ 94

P R O C E S S S Y M B O L SIGMA

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 75 76 77

USE SEGMENTS/ 77

P R O C E S S S Y M B O L SMALL

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 98 105 109

USE SEGMENTS/ 105 109

P R O C E S S S Y M B O L SS

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 94 105 109

USE SEGMENTS/ 94 105 109

P R O C E S S S Y M B O L VEC

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 52 55 65 69

USE SEGMENTS/ 52 55 65 69 99 113

P R O C E S S S Y M B O L Z

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 109

USE SEGMENTS/ 109

SCAN:331

EQF:331

D:

LINES:331 FIELDATA

PATH NUMBER 1

NUMBER OF BRANCHES IN NETWORK = 157
 NUMBER OF BRANCHES PREVIOUSLY TRAVERSED = 5
 NUMBER OF NEW BRANCHES TRAVERSED BY CURRENT PATH = 145
 TOTAL NUMBER OF BRANCHES TRAVERSED = 150
 (TOTAL # TRAVERSED)/(TOTAL BRANCHES) PERCENTAGE = 95
 (NEW BRANCH)/(TOTAL BRANCH) PERCENTAGE = 92

OPTIMAL PATH OF SEGMENTS - -

1 - 2 - 3 - 4 - 6 - 3 - 5 - 6 - 7 - 8 -
 13 - 14 - 18 - 13 - 15 - 16 - 17 - 16 - 17 - 18 -
 19 - 20 - 23 - 19 - 21 - 22 - 21 - 22 - 23 - 24 -
 2 - 3 - 4 - 6 - 7 - 9 - 10 - 9 - 10 - 11 -
 12 - 11 - 12 - 13 - 14 - 18 - 19 - 20 - 23 - 24 -
 25 - 26 - 27 - 26 - 27 - 25 - 26 - 27 - 28 - 29 -
 30 - 29 - 30 - 28 - 29 - 30 - 31 - 32 - 33 - 32 -
 33 - 34 - 35 - 36 - 35 - 36 - 37 - 38 - 40 - 41 -
 43 - 34 - 35 - 36 - 37 - 39 - 40 - 42 - 43 - 44 -
 45 - 47 - 48 - 49 - 48 - 49 - 50 - 51 - 54 - 55 -
 58 - 59 - 60 - 61 - 63 - 64 - 67 - 68 - 71 - 72 -
 79 - 80 - 82 - 83 - 85 - 86 - 87 - 89 - 90 - 91 -
 93 - 94 - 95 - 59 - 60 - 62 - 60 - 62 - 63 - 65 -
 66 - 54 - 55 - 58 - 59 - 60 - 61 - 63 - 64 - 67 -
 68 - 71 - 73 - 74 - 76 - 77 - 78 - 59 - 60 - 61 -
 63 - 64 - 67 - 68 - 71 - 73 - 75 - 77 - 78 - 59 -
 60 - 61 - 63 - 64 - 67 - 68 - 71 - 72 - 79 - 81 -
 85 - 86 - 88 - 90 - 92 - 94 - 95 - 59 - 60 - 61 -
 63 - 64 - 67 - 68 - 71 - 72 - 79 - 80 - 82 - 84 -
 86 - 88 - 90 - 92 - 94 - 95 - 59 - 60 - 61 - 63 -
 65 - 66 - 54 - 56 - 57 - 96 - 90 - 99 - 100 - 101 -
 102 - 101 - 102 - 100 - 101 - 102 - 103 - 104 - 109 - 110 -
 112 - 99 - 100 - 101 - 102 - 103 - 105 - 106 - 108 - 99 -
 100 - 101 - 102 - 103 - 105 - 106 - 107 - 113 - 114 - 115 -
 114 - 115 - 116 -

PATH NUMBER 2

NUMBER OF BRANCHES IN NETWORK = 157
 NUMBER OF BRANCHES PREVIOUSLY TRAVERSED = 150
 NUMBER OF NEW BRANCHES TRAVERSED BY CURRENT PATH = 7
 TOTAL NUMBER OF BRANCHES TRAVERSED = 157
 (TOTAL # TRAVERSED)/(TOTAL BRANCHES) PERCENTAGE = 100
 (NEW BRANCH)/(TOTAL BRANCH) PERCENTAGE = 4

OPTIMAL PATH OF SEGMENTS - -

1 - 2 - 3 - 4 - 6 - 7 - 8 - 13 - 14 - 18 -
 19 - 20 - 23 - 24 - 25 - 26 - 27 - 28 - 29 - 30 -
 31 - 32 - 33 - 34 - 35 - 36 - 37 - 38 - 40 - 41 -
 43 - 44 - 46 - 48 - 49 - 50 - 52 - 53 - 96 - 98 -
 99 - 100 - 101 - 102 - 103 - 104 - 109 - 110 - 111 - 113 -
 114 - 115 - 116 -

IF ALL WENT WELL, YOUR PATHS ARE COMPLETED (IF ERRORS
 OCCURRED, FIN AND CALL AN ATDG PROGRAMMER - YOU CAN RESTART LATER).
 YOU ARE READY TO PROCEED WITH THE GENERATION OF DATA.
 RESPOND WITH : WARD .DATAGEN

V

```

*** PATH NUMBER 1
1 - 2 - 3 - 4 - 5 - 6 - 7 - 8 -
13 - 14 - 15 - 16 - 17 - 18 - 19 - 20 -
19 - 20 - 21 - 22 - 23 - 24 - 25 - 26 -
2 - 3 - 4 - 5 - 6 - 7 - 8 - 9 - 10 - 11 -
12 - 11 - 12 - 13 - 14 - 15 - 16 - 17 - 18 - 19 - 20 - 21 - 22 - 23 - 24 -
25 - 26 - 27 - 28 - 29 - 30 - 31 - 32 - 33 - 34 - 35 -
30 - 29 - 30 - 28 - 29 - 30 - 31 - 32 - 33 - 34 - 35 -
33 - 34 - 35 - 36 - 37 - 38 - 39 - 40 - 41 - 42 -
43 - 34 - 35 - 36 - 37 - 38 - 39 - 40 - 41 - 42 - 43 - 44 -
45 - 47 - 48 - 49 - 50 - 51 - 52 - 53 - 54 - 55 -
58 - 59 - 60 - 61 - 62 - 63 - 64 - 65 - 66 - 67 - 68 - 69 - 70 - 71 - 72 -
79 - 80 - 81 - 82 - 83 - 84 - 85 - 86 - 87 - 88 - 89 - 90 - 91 -
93 - 94 - 95 - 96 - 97 - 98 - 99 - 100 - 101 - 102 - 103 - 104 - 105 - 106 - 107 - 108 - 109 - 110 -
66 - 54 - 55 - 56 - 57 - 58 - 59 - 60 - 61 - 62 - 63 - 64 - 65 - 66 - 67 -
68 - 71 - 72 - 73 - 74 - 75 - 76 - 77 - 78 - 79 - 80 - 81 - 82 - 83 - 84 - 85 - 86 - 87 - 88 - 89 - 90 - 91 -
63 - 64 - 65 - 66 - 67 - 68 - 69 - 70 - 71 - 72 - 73 - 74 - 75 - 76 - 77 - 78 - 79 - 80 - 81 - 82 - 83 - 84 - 85 - 86 - 87 - 88 - 89 - 90 - 91 -
60 - 61 - 62 - 63 - 64 - 65 - 66 - 67 - 68 - 69 - 70 - 71 - 72 - 73 - 74 - 75 - 76 - 77 - 78 - 79 - 80 - 81 - 82 - 83 - 84 - 85 - 86 - 87 - 88 - 89 - 90 - 91 -
85 - 86 - 87 - 88 - 89 - 90 - 91 - 92 - 93 - 94 - 95 - 96 - 97 - 98 - 99 - 100 - 101 - 102 - 103 - 104 - 105 - 106 - 107 - 108 - 109 - 110 -
63 - 64 - 65 - 66 - 67 - 68 - 69 - 70 - 71 - 72 - 73 - 74 - 75 - 76 - 77 - 78 - 79 - 80 - 81 - 82 - 83 - 84 - 85 - 86 - 87 - 88 - 89 - 90 - 91 -
86 - 88 - 90 - 92 - 94 - 95 - 96 - 97 - 98 - 99 - 100 - 101 - 102 - 103 - 104 - 105 - 106 - 107 - 108 - 109 - 110 -
65 - 66 - 54 - 56 - 57 - 58 - 59 - 60 - 61 - 62 - 63 - 64 - 65 - 66 - 67 - 68 - 69 - 70 - 71 - 72 - 73 - 74 - 75 - 76 - 77 - 78 - 79 - 80 - 81 - 82 - 83 - 84 - 85 - 86 - 87 - 88 - 89 - 90 - 91 -
102 - 101 - 102 - 100 - 101 - 102 - 103 - 104 - 105 - 106 - 107 - 108 - 109 - 110 - 111 - 112 - 113 - 114 - 115 -
112 - 99 - 100 - 101 - 102 - 103 - 104 - 105 - 106 - 107 - 108 - 109 - 110 - 111 - 112 - 113 - 114 - 115 -
100 - 101 - 102 - 103 - 104 - 105 - 106 - 107 - 108 - 109 - 110 - 111 - 112 - 113 - 114 - 115 -
114 - 115 - 116 -

```

*** BRANCH EXPRESSIONS WHICH AFFECT EXECUTION

BRANCH EXPRESSION	CONDITION
*****	*****
GREAT.GE.AB	TRUE
I-N	<=ZERO
GREAT.GE.AB	FALSE
I-N	> ZERO
MAX.EQ.K1	TRUE
A(I,J).EQ.0.	TRUE
I-N	<=ZERO
A(I,J).EQ.0.	FALSE
M-N	<=ZERO
M-N	> ZERO
I-N	> ZERO
A(M,J).EQ.0.	TRUE
M-N	<=ZERO
A(N,J).EQ.0.	FALSE
I-N	<=ZERO
I-N	> ZERO
M-N	> ZERO
J-K	<=ZERO
GREAT.GE.AB	TRUE
I-N	> ZERO
MAX.EQ.K1	FALSE
L-N	<=ZERO
L-N	> ZERO
L-N	<=ZERO
L-N	> ZERO
A(I,J).EQ.0.	TRUE
I-N	> ZERO
A(M,J).EQ.0.	TRUE
M-N	> ZERO
J-K	> ZERO
J-N	<=ZERO

J-N	> ZERO
SI-N	<=ZERO
J-N	> ZERO
SI-N	> ZERO
J-L	<=ZERO
J-L	> ZERO
SI-N	<=ZERO
J-L	> ZERO
SI-N	> ZERO
I-N	<=ZERO
I-N	> ZERO
J-N	<=ZERO
J-N	> ZERO
RS.LE.P	TRUE
CS.LE.C	TRUE
I-N	<=ZERO
J-N	> ZERO
RS.LE.P	FALSE
CS.LE.C	FALSE
I-N	> ZERO
R.GE.C	TRUE
I-N	<=ZERO
I-N	> ZERO
N.NE.1	TRUE
N.NE.2	TRUE
ABS(A(N2I,N2I-1)).LT.DELTA	TRUE
L.NE.N	TRUE
(L-N+1).NE.0	TRUE
DISC.GE.0.	TRUE
B.GE.0.	TRUE
P1.NE.0.	TRUE
ABS(A(N,N)-P1).GE.ABS(A(N,N)-P	TRUE
2)	
AP1.NE.0.	TRUE
ABS(A(N2I,N2I-1)).LT.DELTA	FALSE
I-N	<=ZERO
ABS(A(N2I,N2I-1)).LT.DELTA	FALSE
I-N	> ZERO
L.NE.N	FALSE
N.NE.2	TRUE
ABS(A(N2I,N2I-1)).LT.DELTA	TRUE
L.NE.N	TRUE
(L-N+1).NE.0	TRUE
DISC.GE.0.	FALSE
DENOM.NE.0.	TRUE
ABS(A(N2I,N2I-1)).LT.DELTA	TRUE
L.NE.N	TRUE
(L-N+1).NE.0	TRUE
DISC.GE.0.	FALSE
DENOM.NE.0.	FALSE
ABS(A(N2I,N2I-1)).LT.DELTA	TRUE
L.NE.N	TRUE
(L-N+1).NE.0	TRUE
DISC.GE.0.	TRUE
B.GE.0.	FALSE
ABS(A(N,N)-P1).GE.ABS(A(N,N)-P	FALSE
2)	
AP1.NE.0.	FALSE
ABS(A(N2I,N2I-1)).LT.DELTA	TRUE
L.NE.N	TRUE
(L-N+1).NE.0	TRUE
DISC.GE.0.	TRUE
B.GE.0.	TRUE
P1.NE.0.	FALSE
ABS(A(N,N)-P1).GE.ABS(A(N,N)-P	FALSE
2)	

ORIGINAL PAGE IS
OF POOR QUALITY

AP1.NE.0.
 ABCD.N21.N21-100.LT.DELTA
 L.NE.0
 N.NE.2
 NVEC.EQ.0
 J-NN
 J-NN
 &I-NN
 J-NN
 &I-NN
 C2.NE.0.
 NVEC.EQ.NUM
 J-NN
 &I-NN
 C2.NE.0.
 NVEC.EQ.NUM
 J-NN
 &I-NN
 C2.NE.0.
 NVEC.EQ.NUM
 I-NN
 I-NN

FALSE
 TRUE
 FALSE
 FALSE
 FALSE
 <=ZERO
 > ZERO
 <=ZERO
 > ZERO
 > ZERO
 TRUE
 FALSE
 > ZERO
 > ZERO
 FALSE
 FALSE
 > ZERO
 > ZERO
 FALSE
 TRUE
 <=ZERO
 > ZERO

*** PATH NUMBER 2

1 - 2 - 3 - 4 - 5 - 6 - 7 - 8 - 13 - 14 - 18 -
 19 - 20 - 23 - 24 - 25 - 26 - 27 - 28 - 29 - 30 -
 31 - 32 - 33 - 34 - 35 - 36 - 37 - 38 - 40 - 41 -
 43 - 44 - 46 - 48 - 49 - 50 - 52 - 53 - 96 - 98 -
 99 - 100 - 101 - 102 - 103 - 104 - 109 - 110 - 111 - 113 -
 114 - 115 - 116 -

*** BRANCH EXPRESSIONS WHICH AFFECT EXECUTION

BRANCH EXPRESSION

CONDITION

GREAT.GE.AB

TRUE

I-N

> ZERO

MAX.EQ.K1

TRUE

A(I,J).EQ.0.

TRUE

I-N

> ZERO

A(M,J).EQ.0.

TRUE

M-N

> ZERO

J-K

> ZERO

J-N

> ZERO

&I-N

> ZERO

J-L

> ZERO

&I-N

> ZERO

I-N

> ZERO

J-N

> ZERO

PS.LE.F

TRUE

CS.LE.C

TRUE

I-N

> ZERO

R.GE.C

FALSE

I-N

> ZERO

N.NE.1

FALSE

NVEC.EQ.0

FALSE

J-NN

> ZERO

&I-NN

> ZERO

C2.NE.0.

TRUE

NVEC.EQ.NUM

TRUE

I-NN

> ZERO

SKLTON END OF JOB

ID,R 9.

EAD-ONLY MODE

EASE UPPER ASSUMED

J 14.01-06/03-19:05-(D,)

JIT

:

*** INPUT VARIABLES WHICH AFFECT BRANCHING

REC A

*** OTHER INPUT VARIABLES

EIGVEC Z BB ABS CC SS

*** VARIABLES WHICH ARE COMPUTED

I JI A NMAX C1 SMALL NN NUM

EIGVEC NVEC CC SS J C2 Z BB

NM DEL VEC IJK N N2I L SHIFT

ERR AR2 R2 R1 DISC C B RHO

SIGMA AR4 AR3 DENOM ANUM DELTA R CS

KII KIJ R5 K1 D M E AB

MAX GREAT K2 K

*** SKELETON LISTING OF THE CANDIDATE PATH FOR HSMAN

SEG 1 SEG 2 STATEMENT

```

      .
      .
1      NN = 4
1      NVEC = 4
1      N=NN
1      K=N-2
2      DO1J=1,K
2      K1=J+1
2      K2=J+2
2      GREAT= ABS(A(K1,J))
2      MAX=K1
3      DO2I=K2,N
3      AB= ABS(A(I,J))
3      4      IF (GREAT.GE.AB) GO TO 2
      .
5      MAX=I
6      2 CONTINUE
7      INT(J)=MAX
7      8      IF (MAX.EQ.K1) GO TO 5
9      4 DO6L=J,N
9      E=A(K1,L)
      .
10     6 A(MAX,L)=E
11     DO7L=1,N
11     E=A(L,K1)
      .
12     7 A(L,MAX)=E
13     5 DO8I=K2,N
13     14     IF (A(I,J).EQ.0.) GO TO 8
15     A(I,J)=-A(I,J)/A(K1,J)
16     DO11M=K1,N
17     11 A(I,M)=A(I,M)+A(I,J)*A(K1,M)
18     8 CONTINUE
19     DO 2000 M=K2,N
19     20     IF (A(M,J).EQ.0.) GO TO 2000
21     10 DO 2001 I=1,N
22     2001 A(I,K1)=A(I,K1)-A(I,M)*A(M,J)

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

23      2000 CONTINUE
24      1 CONTINUE
25      DO 105 I=1,N
26      DO 105 J=1,N
27      105 D(I,J)=A(I,J)
28      DO 106 I=3,N
28      L=I-2
29      DO 106 J=1,L
30      106 A(I,J)=0.0
31      K1=N+1
31      R=0.0
31      C=0.0
32      DO33 I=1,N
32      R=R+ ABS(A(1,I))
33      33 C=C+ ABS(A(I,N))
34      DO34 I=2,N
34      RS=0.0
34      CS=0.0
34      L=I-1
35      DO35 J=L,N
35      RS=RS+ ABS(A(I,J))
35      K1J=K1-J
35      K1I=K1-I
36      35 CS=CS+ ABS(A(K1J,K1I))
37      38 IF (RS.LE.R) GO TO 36
39      37 R=RS
40      41 36 IF (CS.LE.C) GO TO 34
42      44 C=CS
43      34 CONTINUE
      .
44      45 IF (R.GE.C) GO TO 54
      .
47      54 DELTA=C*1.0E-11
47      DEL=C/17.1829
48      153 DO 107 I=1,N
49      107 A(I,I)=A(I,I)-DEL
50      51 55 IF (N.NE.1) GO TO 46
      .
54      55 48 IF (N.NE.2) GO TO 80
56      49 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
57      GO TO 76
58      80 AR1=0.0
58      AR3=0.0
58      AR4=0.0
59      61 L=N
60      DO56 I=2,N
60      N2I=N+2-I
60      61 IF (ABS(A(N2I,N2I-1)).LT.DELTA) GO TO 57
62      56 L=L-1
63      64 57 IF (L.NE.N) GO TO 59
65      58 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
66      N=N-1
66      GO TO 48
67      68 59 IF ((L-N+1).NE.0) GO TO 75
      .

```

```

71      75 B=A(N-1,N-1)+A(N,N)
71      C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
71      C1=A(N-1,N-1)-A(N,N)
71      DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)
71      72 IF (DISC.GE.0.) GO TO 63

```

```

73      DENOM=AR3*AR3+AR4*AR4
73      74 IF (DENOM.NE.0.) GO TO 200

```

```

75      GO TO 203
76      200 RHO=C

```

```

77      203 AR3=R1

```

```

77      CALL COMPIT(A,NMAX,L,N,RHO,SIGMA)

```

```

78      GO TO 61
79      80 63 IF (B.GE.0.) GO TO 65

```

```

81      64 R1=.5*(B-SQRT(DISC))
81      GO TO 66

```

```

82      65 R1=.5*(B+SQRT(DISC))
82      83 IF (R1.NE.0.) GO TO 66

```

```

84      83 R2=0.0
84      GO TO 84

```

```

85      66 R2=C/R1
86      87 84 IF (ABS(A(N,N)-R1).GE. ABS(A(N,N)-R2))GO TO 68

```

```

88      GO TO 69

```

```

89      68 AR2=R2
90      91 69 IF (AR1.NE.0.) GO TO 71

```

```

92      GO TO 73
93      71 SHIFT=AR2

```

```

94      73 AR1=AR2

```

```

94      CALL REALIT(A,NMAX,L,N,SHIFT,CC,SS)
95      GO TO 61

```

```

96      97 76 IF (NVEC.EQ.0) GO TO 600

```

```

98      NUM=0
98      JI=1

```

```

99      513 C1=VEC(JI)

```

```

99      C2=VEC(JI+1)

```

```

99      NUM=NUM+1

```

```

100     DO 570 I=1,NN

```

```

101     DO 570 J=1,NN

```

```

102     570 A(I,J)=D(I,J)

```

```

103     104 IF (C2.NE.0.) GO TO 561

```

```

105     CALL RVEC(A,NMAX,INT,C1,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS)

```

```

106     JI=JI+2

```

```

106     107 IF (NVEC.EQ.NUM)GO TO 600

```

```

108     GO TO 513

```

```

109     561 CALL IMVEC(A,NMAX,INT,C1,C2,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS)

```

```

110     JI=JI+4

```

```

110     111 IF (NVEC.EQ.NUM)GO TO 600

```

Z,BR+

VI

```

112          GO TO 513
113          600 CONTINUE
          .
114          DO 20 I=1,NN
115          20  WRITE(6,700) (EIGVEC(I,J),J=1,NN)
*** INPUT VARIABLES WHICH AFFECT BRANCHING
VEC      A
*** OTHER INPUT VARIABLES
EIGVEC CC  SS  Z  BB
*** VARIABLES WHICH ARE COMPUTED
I  JI  A  NMAX  C1  C2  SMALL  NN
VUM  EIGVEC NVEC  CC  SS  Z  BB  J
VM  DEL  VEC  IJK  N  DELTA  CS  K11
K1J  RS  L  C  R  K1  D  M
ZB  MAX  GREAT  K2  K
*** SKELETON LISTING OF THE CANDIDATE PATH FOR HSMMAIN
SEG 1  SEG 2  STATEMENT

```

```

1          NN = 4
1          NVEC = 4
1          N=NN
1          K=N-2
2          DO1J=1,K
2          K1=J+1
2          K2=J+2
2          GREAT= ABS(A(K1,J))
2          MAX=K1
3          DO2I=K2,N
3          AB= ABS(A(I,J))
3          4  IF (GREAT.GE.AB) GO TO 2
          .
6          2  CONTINUE
7          INT(J)=MAX
7          8  IF (MAX.EQ.K1) GO TO 5
          .
13         5  DO8I=K2,N
13         14 IF (A(I,J).EQ.D.) GO TO 8
          .
18         8  CONTINUE
19         DO 2000 M=K2,N
19         20 IF (A(M,J).EQ.D.) GO TO 2000
          .
23         2000 CONTINUE
24         1  CONTINUE
25         DO 105 I=1,N
26         DO 105 J=1,N
27         105 D(I,J)=A(I,J)
28         DO 106 I=3,N
28         L=I-2
29         DO 106 J=1,L
30         106 A(I,J)=D.D
31         K1=N+1

```

```

31      R=0.0
31      C=0.0
32      DO33 I=1,N
32      R=R+ ABS(A(1,I))
33      33 C=C+ ABS(A(I,N))
34      DO34 I=2,N
34      RS=0.0
34      CS=0.0
34      L=I-1
35      DO35 J=L,N
35      RS=RS+ ABS(A(I,J))
35      K1J=K1-J
35      K1I=K1-I
36      35 CS=CS+ ABS(A(K1J,K1I))
37      38      IF (RS.LE.R) GO TO 36
      .
40      41      36 IF (CS.LE.C) GO TO 34
      .
43      34 CONTINUE
      .
44      45      IF (R.GE.C) GO TO 54
      .
46      GO TO 153
      .
48      153 DO 107 I=1,N
49      107 A(I,I)=A(I,I)-DEL
50      51      55 IF (N.NE.1) GO TO 48
52      47 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
53      GO TO 76
      .
96      97      76 IF (NVEC.EQ.0) GO TO 600
      .
98      NUM=0
98      JI=1
      .
99      C2=VEC(JI+1)
99      NUM=NUM+1
100     DO 570 I=1,NN
101     DO 570 J=1,NN
102     570 A(I,J)=D(I,J)
103     104     IF (C2.NE.0.) GO TO 561
      .
109     561 CALL INVEC(A,NMAX,INT,C1,C2,SMALL,NN,NUM,EIGVEC,NVEC,CC,S
110     111     IF (NVEC.EQ.NUM) GO TO 600
      .
113     600 CONTINUE
      .
114     DO 20 I=1,NN
115     20 WRITE(6,700) (EIGVEC(I,J),J=1,NN)
CAN:277
OF:277

```


COMPIT

B, J 14.

SE UPPER ASSUMED

14.01-06/04-14:00-(0,1)

IT

SEGMENT
NUMBERS

FORTTRAN STATEMENTS

```

      ELEMENT COMPT
      SUBROUTINE COMPT(A,NMAX,L,N,RHO,SIGMA)
      DIMENSION A(NMAX,NMAX)
      M=N-1
      DO 1 I=L,M
      2, 3   IF (I.NE.L) GO TO 3
      4     2 B1=A(I,I)*(A(I,I)-SIGMA)+A(I,I+1)*A(I+1,I)+R
      *H0
      B2=A(I+1,I)*(A(I,I)+A(I+1,I+1)-SIGMA)
      B3=A(I+1,I)*A(I+2,I+1)
      A(I+2,I)=0.0
      GO TO 6
      5     3 B1=A(I,I-1)
      B2=A(I+1,I-1)
      5, 6   IF (I.EQ.M) GO TO 5
      7     4 B3=A(I+2,I-1)
      GO TO 6
      8     5 B3=0.0
      9     6 D= SQRT(B1*B1+B2*B2+B3*B3)
      9, 10  IF (B1.GE.0.) GO TO 6
      11    7 D=-D
      12, 13  8 IF (D.EQ.0.) GO TO 1
      14    9 C1=B2/(D+B1)
      C2=B3/(D+B1)
      E=2.0/(1.0+C1*C1+C2*C2)
      DO 10 K=I,N
      15, 16  IF (I.EQ.M) GO TO 12
      17    11 G=E*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))
      A(I,K)=A(I,K)-G
      A(I+1,K)=A(I+1,K)-C1*G
      A(I+2,K)=A(I+2,K)-C2*G
      GO TO 10
      18    12 G=E*(A(I,K)+C1*A(I+1,K))
      A(I,K)=A(I,K)-G
      A(I+1,K)=A(I+1,K)-C1*G
      19    13 CONTINUE
      20, 21  IF (I.EQ.L) GO TO 14
      22    13 A(I,I-1)=0
      A(I+1,I-1)=0.0
      22, 23  IF (I.EQ.M) GO TO 14
      24    15 A(I+2,I-1)=0.0
      25    14 J=I+2
      25, 26  IF (J.LE.N) GO TO 17
      27    16 J=N
      28    17 DO 18 K=L,J
      28, 29  IF (I.EQ.M) GO TO 20
```

ORIGINAL PAGE IS
OF POOR QUALITY

H

30		19	$G = E * (A(K, I) + C1 * A(K, I+1) + C2 * A(K, I+2))$
30			$A(K, I) = A(K, I) - G$
30			$A(K, I+1) = A(K, I+1) - C1 * G$
32			$A(K, I+2) = A(K, I+2) - C2 * G$
30			GO TO 18
31		20	$G = E * (A(K, I) + C1 * A(K, I+1))$
31			$A(K, I) = A(K, I) - G$
31			$A(K, I+1) = A(K, I+1) - C1 * G$
32		18	CONTINUE
33			J=I+3
33, 34			IF (J.GT.N) GO TO 1
35		21	$G = E * C2 * A(I+3, I+2)$
35			$A(I+3, I) = -G$
35			$A(I+3, I+1) = -C1 * G$
35			$A(I+3, I+2) = A(I+3, I+2) - C2 * G$
36		1	CONTINUE
37			RETURN
0			END

SEGMENT
NUMBERS

FORTRAN STATEMENTS

0 ELEMENT EOF
24:77
0F:77

INES:77 FIELDATA

STATIC ERROR ANALYSIS FOR ELEMENT COMPIT

***CHECK FOR INFINITE LOOPS

NO INFINITE LOOPS HAVE BEEN DETECTED

***CHECK FOR UNREACHABLE CODE

NO UNREACHABLE CODE HAS BEEN DETECTED

***CHECK SET/USE OF INTERNAL SYMBOLS

NO VIOLATIONS HAVE BEEN DETECTED FOR THE FOLLOWING SYMBOLS

B1	B2	B3	C1	C2	D	E
G	I	J	K	M		

STATIC ERROR ANALYSIS IS COMPLETE.

ORIGINAL PAGE IS
OF POOR QUALITY

III

7, J 11.

SE UPPER ASSUMED

14.01-06/04-14:01-(0,1)

II

****CHECK SET/USE OF INTERNAL SYMBOLS**

P R O C E S S S Y M B O L B1

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 4 5

USE SEGMENTS/ 9 14

P R O C E S S S Y M B O L B2

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 4 5

USE SEGMENTS/ 9 14

P R O C E S S S Y M B O L B3

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 4 7 8

USE SEGMENTS/ 9 14

P R O C E S S S Y M B O L C1

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 14

USE SEGMENTS/ 14 17 18 30 31 35

P R O C E S S S Y M B O L C2

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 14

USE SEGMENTS/ 14 17 30 35

P R O C E S S S Y M B O L D

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 9

USE SEGMENTS/ 11 12 14 22

P R O C E S S S Y M B O L E

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 14

USE SEGMENTS/ 17 18 30 31 35

P R O C E S S S Y M B O L G

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 17 18 30 31 35

USE SEGMENTS/ 17 18 30 31 35

III

PROCESS SYMBOL I

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 2

USE SEGMENTS/ 2 5 7 15 17 18 20 22 24

25 28 30 31 33 35

PROCESS SYMBOL J

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 25 27 33

USE SEGMENTS/ 25 28 33

PROCESS SYMBOL K

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 15 28

USE SEGMENTS/ 17 18 30 31

PROCESS SYMBOL M

NO SET VIOLATIONS HAVE BEEN DETECTED

NO USE VIOLATIONS HAVE BEEN DETECTED

SET SEGMENTS/ 1

USE SEGMENTS/ 2 5 15 22 28

AH:76

F:76

NES:76 FIELDATA

27A

PATH NUMBER 1

IV

NUMBER OF BRANCHES IN NETWORK = 49
 NUMBER OF BRANCHES PREVIOUSLY TRAVERSED = 0
 NUMBER OF NEW BRANCHES TRAVERSED BY CURRENT PATH = 46
 TOTAL NUMBER OF BRANCHES TRAVERSED = 46
 (TOTAL # TRAVERSED)/(TOTAL BRANCHES) PERCENTAGE = 93
 (NEW BRANCH)/(TOTAL BRANCH) PERCENTAGE = 93

OPTIMAL PATH OF SEGMENTS - -

1 - 2 - 4 - 9 - 10 - 12 - 14 - 15 - 17 - 19 -
 15 - 17 - 19 - 20 - 21 - 25 - 26 - 28 - 30 - 32 -
 28 - 30 - 32 - 33 - 35 - 36 - 2 - 3 - 5 - 7 -
 9 - 11 - 12 - 14 - 15 - 17 - 19 - 20 - 22 - 24 -
 25 - 26 - 28 - 30 - 32 - 33 - 34 - 36 - 2 - 3 -
 5 - 6 - 8 - 9 - 10 - 12 - 14 - 15 - 16 - 18 -
 19 - 20 - 22 - 23 - 25 - 27 - 28 - 29 - 31 - 32 -
 33 - 34 - 36 - 37 -

PATH NUMBER 2

NUMBER OF BRANCHES IN NETWORK = 49
 NUMBER OF BRANCHES PREVIOUSLY TRAVERSED = 46
 NUMBER OF NEW BRANCHES TRAVERSED BY CURRENT PATH = 3
 TOTAL NUMBER OF BRANCHES TRAVERSED = 49
 (TOTAL # TRAVERSED)/(TOTAL BRANCHES) PERCENTAGE = 100
 (NEW BRANCH)/(TOTAL BRANCH) PERCENTAGE = 6

OPTIMAL PATH OF SEGMENTS - -

1 - 2 - 4 - 9 - 10 - 12 - 13 - 36 - 37 -

*** PATH NUMBER 1
 1 - 2 - 4 - 9 - 10 - 12 - 14 - 15 - 17 - 19 -
 15 - 17 - 19 - 20 - 21 - 25 - 26 - 28 - 30 - 32 -
 28 - 30 - 32 - 33 - 35 - 36 - 2 - 3 - 5 - 7 -
 9 - 11 - 12 - 14 - 15 - 17 - 19 - 20 - 22 - 24 -
 25 - 26 - 28 - 30 - 32 - 33 - 34 - 36 - 2 - 3 -
 5 - 6 - 8 - 9 - 10 - 12 - 14 - 15 - 16 - 18 -
 19 - 20 - 22 - 23 - 25 - 27 - 28 - 29 - 31 - 32 -
 33 - 34 - 36 - 37 -

*** BRANCH EXPRESSIONS WHICH AFFECT EXECUTION

BRANCH EXPRESSION *****	CONDITION *****
I.NE.L	FALSE
B1.GE.0.	TRUE
D.EQ.0.	FALSE
I.EQ.M	FALSE
K-N	<=ZERO
I.EQ.M	FALSE
K-N	> ZERO
I.EQ.L	TRUE
J.LE.N	TRUE
I.EQ.M	FALSE
K-J	<=ZERO
I.EQ.M	FALSE
K-J	> ZERO
J.GT.N	FALSE
I-M	<=ZERO
I.NE.L	TRUE
I.EQ.M	FALSE
B1.GE.0.	FALSE
D.EQ.0.	FALSE
I.EQ.M	FALSE
K-N	> ZERO
I.EQ.L	FALSE
I.EQ.M	FALSE
J.LE.N	TRUE
I.EQ.M	FALSE
K-J	> ZERO
J.GT.N	TRUE
I-M	<=ZERO
I.NE.L	TRUE
I.EQ.M	TRUE
B1.GE.0.	TRUE
D.EQ.0.	FALSE
I.EQ.M	TRUE
K-N	> ZERO
I.EQ.L	FALSE
I.EQ.M	TRUE
J.LE.N	FALSE
I.EQ.M	TRUE
K-J	> ZERO
J.GT.N	TRUE
I-M	> ZERO

*** PATH NUMBER 2
 1 - 2 - 4 - 9 - 10 - 12 - 13 - 36 - 37 -
 *** BRANCH EXPRESSIONS WHICH AFFECT EXECUTION

BRANCH EXPRESSION *****	CONDITION *****
I.NE.L	FALSE
B1.GE.0.	TRUE
D.EQ.0.	TRUE
I-M	> ZERO

SED,R 9.

READ=ONLY MODE

CASE UPPER ASSUMED

ED 14.01.06/07-15:44-(0.)

EDIT

0:

1*** INPUT VARIABLES WHICH AFFECT BRANCHING

K L SIGMA A RHO

0*** OTHER INPUT VARIABLES

0*** VARIABLES WHICH ARE COMPUTED

J A G K E C2 C1 D
B3 B2 B1 I M

1*** SKELETON LISTING OF THE CANDIDATE PATH FOR COMPT

SEG 1 SEG 2 STATEMENT

```

1          M=N-1
2          DO 1 I=L,N
2          3      IF (I.NE.L) GO TO 3
4          2      B1=A(L,L)*(A(L,L)-SIGMA)+A(L,L+1)*A(I+1,L)+RHO
4          B2=A(L+1,L)*(A(L,L)+A(L+1,L+1)-SIGMA)
4          B3=A(L+1,L)*A(L+2,L+1)
4          A(L+2,L)=U.U
4          GO TO 6
5          3      B1=A(I,I-1)
5          B2=A(I+1,I-1)
5          6      IF (I.EQ.N) GO TO 5
7          4      B3=A(I+2,I-1)
7          GO TO 6
8          5      B3=U.U
9          6      D=SQRT(B1*B1+B2*B2+B3*B3)
9          10      IF (B1.GE.0.) GO TO 8
11         7      D=-D
12         13      IF (D.EQ.U.) GO TO 1
14         9      C1=B2/(D+B1)
14         C2=B3/(D+B1)
14         E=2.U/(1.U+C1.C1+C2.C2)
15         DO 10 K=1,N
15         16      IF (I.EQ.N) GO TO 12
17         11      G=E*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))
17         A(I,K)=A(I,N)-G
17         A(I+1,K)=A(I+1,K)-C1*G
17         A(I+2,K)=A(I+2,K)-C2*G
17         GO TO 10
18         12      G=E*(A(I,K)+C1*A(I+1,K))
19         10      CONTINUE
20         21      IF (I.EQ.L) GO TO 14
22         23      IF (I.EQ.N) GO TO 14
24         15      A(I+2,I-1)=U.U
25         14      J=I+2
25         26      IF (J.LE.N) GO TO 17
27         16      J=N
28         17      DO 18 K=L,J

```

VI

```

28      29      IF (I.EQ.M) GO TO 20
30      19 G=E*(A(K,I)+C1*A(K,I+1)+C2*A(K,I+2))
30      A(K,I)=A(K,I)-G
30      A(K,I+1)=A(K,I+1)-C1*G
30      A(K,I+2)=A(K,I+2)-C2*G
30      GO TO 18
31      20 G=E*(A(K,I)+C1*A(K,I+1))

```

```

32      18 CONTINUE
33      J=I+3
32      21 IF (J.GT.N) GO TO 1
35      21 G=E*C2*A(I+3,I+2)
35      A(I+3,I+1)=-C1*G
35      A(I+3,I+2)=A(I+3,I+2)-C2*G
36      1 CONTINUE
37      RETURN

```

1*** INPUT VARIABLES WHICH AFFECT BRANCHING

N SIGMA RHO A L

0*** OTHER INPUT VARIABLES

0*** VARIABLES WHICH ARE COMPUTED

D A B3 B2 B1 I M

1*** SKELETON LISTING OF THE CANDIDATE PATH FOR COMPUT

SEG 1 SEG 2 STATEMENT

```

1      M=N-1
2      DO 1 I=L,M
2      3 IF (I.NE.L) GO TO 3
4      2 B1=A(L,L)*(A(L,L)-SIGMA)+A(L,L+1)*A(L+1,L)+RHO
4      B2=A(L+1,L)*(A(L,L)+A(L+1,L+1)-SIGMA)
4      B3=A(L+1,L)*A(L+2,L+1)
4      GO TO 6

```

```

9      6 D=SQRT(B1*B1+B2*B2+B3*B3)
9      10 IF (B1.GE.0.) GO TO 8

```

```

12     13 8 IF (D.EQ.0.) GO TO 1

```

```

36     1 CONTINUE
37     RETURN

```

STATEMENTS	*** FORTUNE FOR IBM/360 ***	EXECUTIONS	TIME	TRUE	PAGE 1
C FOR DESCRIPTION OF METHOD USED IN THIS PROGRAM, SEE **THE QR C TRANSFORMATION** BY FRANCIS, COMPUTER JOURNAL, VOL. 4 PP, 265,332 C START HERE TO PRODUCE UPPER HESSENBERG MATRIX					
DIMENSION A(10,10),VEC(8),EIGVEC(4,4),INT(10),CC(10),SS(10) DIMENSION Z(10),D(10,10),BR(10,10) DATA A / 0.1,5.8,1.9,3.2,6*0.0,2.8,3.9,0.7,0.1,6*0.0, Z 6.2,3.5,0.6,1.2,6*0.0,9.9,6.2,1.7,3.8 /					
NMAX = 10		1	2		
NN = 4		1	2		
NVEC = 4		1	2		
N = NN		1	2		
K = N-2		1	5		
DO 1J=1,K		1	18		
K1=J+1		2	10		
K2=J+2		2	10		
GHFAT= ABS(A(K1,J))		2	24		
HAY=K1		2	4		
DO 1I=K2,N		2	27		
AB= ABS(A(I,J))		3	36		
IF (GREAT.GE,AB) GO TO 2		3	19	2	
3 GREAT=AB		1	2		
HAY=I		1	2		
2 CONTINUE		3			
C INT(J) GIVES INFORMATION ABOUT INTERCHANGES					
INT(J)=MAX		2	8		
IF (MAX.EQ,K1) GO TO 5		2	12	1	
4 DO 1L=J,N		1	27		
E=A(K1,L)		3	21		
A(K1,L)=A(MAX,L)		3	33		
6 A(MAX,L)=E		3	21		
DO 7L=1,N		1	36		
E=A(L,K1)		4	28		
A(L,K1)=A(L,MAX)		4	44		
7 A(L,MAX)=E		4	28		
C INTERCHANGE DONE					
5 DO 1I=K2,N		2	27		
IF (A(I,J).EQ,0.) GO TO 8		3	27	**0*	
A(I,J)=A(I,J)/A(K1,J)		3	78		
DO 11M=K1,N		3	72		
11 A(I,M)=A(I,M)+A(I,J)*A(K1,M)		8	216		
8 CONTINUE		3			
C LEFT MULT. DONE					
DO 2000 M=K2,N		2	27		
IF (A(M,J).EQ,0.) GO TO 2000		3	27	**0*	
10 DO 2001 I=1,N		3	108		
2001 A(I,K1)=A(I,K1)+A(I,M)*A(M,J)		12	324		
2000 CONTINUE		3			
1 CONTINUE		2			
C RIGHT MULT. DONE					
C HESSENBERG MATRIX NOW FORMED AS A(I,J). MULTIPLIERS STORED WITH IT					
DO 105 I=1,N		1	36		
DO 105 J=1,N		4	144		
105 D(I,J)=A(I,J)		16	176		

ORIGINAL PAGE IS
OF POOR QUALITY

PRECEDING PAGE BLANK NOT FILMED

STATE	S	*** FORTUNE FOR IBM/360 ***	EXECUTIONS	TIME	TRUE	P	2
C PUT APPROPRIATE ZEROS WHERE MULTIPLIERS WERE STORED							
DD 106 I=3,N			1	18			
L=I-2			2	10			
DD 106 J=1,L			2	27			
106 A(I,J)=0,0			3	21			
K1=N+1			1	5			
R=0,0			1	2			
C=0,0			1	2			
DD 33 I=1,N			1	36			
R=R+ABS(A(I,I))			4	56			
33 C=C+ABS(A(I,N))			4	56			
DD 44 I=2,N			1	27			
RS=0,0			3	6			
CS=0,0			3	6			
L=I-1			3	15			
DD 55 J=L,N			3	81			
RS=RS+ABS(A(I,J))			9	126			
K1J=K1-J			9	45			
K1I=K1-1			9	45			
35 CS=CS+ABS(A(K1J,K1I))			9	126			
IF (RS.LE.R) GO TO 36			3	21		3	
37 R=RS			**0*				
36 IF (CS.LE.C) GO TO 34			3	19		2	
44 C=C			1	2			
34 CONTINUE			3				
IJK=1			1	2			
IF (P.GE.C1) GO TO 54			1	7		1	
C DELTA IS CRITERION FOR STOPPING AN ITERATION							
53 DELTA=R*1,DE=11			**0*				
C DEL IS ARBITRARY SHIFT							
DEI=R/17,1A29			**0*				
GO TO 153			**0*				
54 DELTA=C*1,DE=11			1	9			
DEI=C/17,1A29			1	10			
C DELTA COMPUTED							
C PRODUCE A-PI, ITERATE ON THIS TO BREAK UP EIGENVALUES OF EQUAL MODULUS							
153 DD 107 I=1,N			1	36			
107 A(I,I)=A(I,I)-DEL			4	56			
55 IF (N.NE.1) GO TO 48			1	7		1	
C SUBROUTINE CALCEV CALCULATES EIGENVALUES OF 1X1 OR 2X2 MATRICES.							
C PI AS ADDED BACK IN CALCEV							
47 CALL CALCEV(A,NHAX,DEL,VEC,IJK,N,1)			**0*				
GO TO 76			**0*				
48 IF (N.NE.2) GO TO 40			3	19		2	
49 CALL CALCEV(A,NHAX,DEL,VEC,IJK,N,2)			1	21			
GO TO 76			1	2			
C OR TRANSFORMATION STARTS HERE							
C INITIALIZE FOR DETERMINING SUBSEQUENT SHIFTS							
80 AR1=0,0			2	4			
AR3=0,0			2	4			
AR4=0,0			2	4			
C RE-ENTER AFTER REALIT OR COMPT SUBROUTINES CALLED							
C SELECT FIRST (I=ZERO) SUB-DIAGONAL ELEMENT FROM BOTTOM							
61 LEN			9	18			
DD 56 I=2,N			9	180			

JTF		*** RUNTIME FOR IBM/360 ***	EXECUTIONS	TIME	TRUE	1	3
	N2I=N+2-1		20	140			
	IF (ABS(A(N2I,N2I-1)).LT.DELTA) GO TO 57		20	344	2		
54	L=L+1		18	90			
C VALUE OF L SELECTED							
57	IF (L.NE.N) GO TO 59		9	59	7		
58	CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)		2	42			
	N=N+1		2	10			
	GO TO 48		2	4			
59	IF ((L-N+1).NE.0) GO TO 75		7	84	7		
60	CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)		**0*				
	N=N+2		**0*				
	GO TO 55		**0*				
C DETERMINE EIGENVALUES OF 2X2 MATRIX == ROW 1 * A(N-1,N-1) A(N-1,N)							
C ROW 2 * A(N,N-1) A(N,N)							
C AND BASE SHIFT ON THESE							
75	R=A(N-1,N-1)+A(N,N)		7	133			
	C=A(N-1,N-1)+A(N,N)-A(N,N-1)+A(N-1,N)		7	294			
	C1=A(N-1,N-1)-A(N,N)		7	133			
	DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)		7	224			
	IF (DISC.GE.0.) GO TO 63		7	49	7		
62	R1=0.5*B		**0*				
	R2=.5*SQRT(ABS(DISC))		**0*				
	ANUM=(R1-AR3)*(R1-AR3)+(R2-AR4)*(R2-AR4)		**0*				
	DENOM=AR3*AR3+AR4*AR4		**0*				
	IF (DENOM.NE.0.) GO TO 200		**0*				
201	RHO=0.0		**0*				
	SIGMA=0.0		**0*				
	GO TO 203		**0*				
200	RHO=C		**0*				
	SIGMA=B		**0*				
203	AR=R1		**0*				
	AR=R2		**0*				
	CALL COMPT(A,NMAX,L,N,RHO,SIGMA)		**0*				
	GO TO 61		**0*				
63	IF (B.GE.0.) GO TO 65		7	35	**0*		
64	R1=.5*(B+SQRT(DISC))		7	497			
	GO TO 66		7	14			
65	R1=.5*(B+SQRT(DISC))		**0*				
	IF (R1.NE.0.) GO TO 66		**0*				
83	R2=0.0		**0*				
	GO TO 84		**0*				
66	R2=C/R1		7	70			
C ROOTS OF 2X2 MATRIX FOUND							
84	IF (ABS(A(N,N)-R1).GE. ABS(A(N,N)-R2)) GO TO 68		7	209	3		
67	AR=R1		4	8			
	GO TO 69		4	8			
68	AR=R2		3	6			
69	IF (AR1.NE.0.) GO TO 71		7	45	5		
72	SHIFT=0.0		2	4			
	GO TO 73		2	4			
71	SHIFT=AR2		5	10			
73	AR1=AR2		7	14			
	CALL REALIT(A,NMAX,L,N,SHIFT,CC,BS)		7	147			
	GO TO 61		7	14			
C ARE VECTORS NEEDED							

ORIGINAL PAGE IS
OF POOR QUALITY

STATEMENTS	*** FORTUNE FOR IBH/360 ***	EXECUTIONS	TIME	TRUE	2 4
76 IF (NVEC.EQ.0) GO TO 600		1	5	**0*	
C INV*RSF ITERATION IS USED FOR EIGENVECTORS					
550 SMALL*DELTA+1.0E-30		1	9		
NUM=0		1	2		
JI=1		1	2		
NN=NN+NN		1	5		
C C1 IS REAL PART OF EIGENVALUE					
513 C1=VEC(JI)		4	16		
C C2 IS COMPLEX PART OF EIGENVALUE					
C2=VEC(JI+1)		4	28		
NUM=NUM+1		4	20		
C REPLACE HESSENBERG MATRIX AND MULTIPLIERS					
DO 570 I=1,NN		4	144		
DO 570 J=1,NN		16	576		
570 A(I,J)=D(I,J)		64	704		
IF (C2.NE.0.) GO TO 561		4	20	**0*	
CALL RVEC(A,NMAX,INT,C1,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS)		4	124		
JI=JI+2		4	20		
IF(NVEC.EQ.NUM)GO TO 600		4	22	1	
GO TO 513		3	6		
561 CALL IMVEC(A,NMAX,INT,C1,C2,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS,2,BB)		**0*			
C NEXT STATEMENT IGNORES CONJUGATE COMPLEX EIGENVALUES					
JI=JI+4		**0*			
IF(NVEC.EQ.NUM)GO TO 600		**0*			
GO TO 513		**0*			
600 CONTINUE		1			
WRITE(6,700) (VEC(I),I=1,NN)		1	144		
DO 20 I=1,NN		1	36		
20 WRITE(6,700) (EIGVEC(I,J),J=1,NN)		4	664		
700 FORMAT(1H,5E20.8 / 5E20.8)					
END					

JAT	S	*** FORTUNE FOR IBM/360 ***	EXECUTIONS	TIME	TRUE	E 5
		SUBROUTINE CALCEV(A,NMAX,DEL,VEC,IJK,N,M)	3	132		
		DIMENSION A(NMAX,NMAX),VEC(1)				
		IF (M.NE.1) GO TO 2	3	17	1	
1		AI=0.0	2	4		
		A(N,M)=A(N,M)+DEL	2	28		
		C STORE EIGENVALUES IN VEC				
		VEC(IJK)=A(N,M)	2	18		
		VEC(IJK+1)=AI	2	14		
		IJK=IJK+2	2	10		
		RETURN	2	4		
2		B=A(N-1,N-1)+A(N,N)	1	19		
		C=A(N-1,N-1)+A(N,N)-A(N,N-1)+A(N-1,N)	1	42		
		C1=A(N-1,N-1)-A(N,N)	1	19		
		DISC=C1+C1+4.0+A(N,N-1)+A(N-1,N)	1	32		
		IF (DISC.LT.0.) GO TO 9	1	5	**0*	
4		AI=0.0	1	2		
		IF (B.GE.0.) GO TO 6	1	7	1	
		R1=0.5*(B+SQRT(DISC))	**0*			
		GO TO 7	**0*			
6		R1=0.5*(B+SQRT(DISC))	1	71		
		IF (R1.NE.0.) GO TO 7	1	7	1	
10		R2=0.0	**0*			
		GO TO 11	**0*			
7		R2=C/R1	1	10		
11		R1=R1+DEL	1	5		
		R2=R2+DEL	1	5		
		VEC(IJK)=R2	1	4		
		VEC(IJK+1)=AI	1	7		
		VEC(IJK+2)=R1	1	7		
		VEC(IJK+3)=AI	1	7		
		GO TO 8	1	2		
9		R1=0.5*B	**0*			
		AI=-.5*SQRT(ABS(DISC))	**0*			
		R1=R1+DEL	**0*			
		VEC(IJK)=R1	**0*			
		VEC(IJK+1)=-AI	**0*			
		VEC(IJK+2)=R1	**0*			
		VEC(IJK+3)=AI	**0*			
8		IJK=IJK+4	1	5		
		RETURN	1	2		
		END				

ORIGINAL PAGE IS
OF POOR QUALITY

STATE	TS	*** FORTUNE FOR IBM/360 ***	EXECUTIONS	TIME	TRUE	I	6
		SUBROUTINE REALIT(A,NMAX,L,N,SHIFT,C,S)	7	308			
		DIMENSION A(NMAX,NMAX),C(1),S(1)					
		IF (SHIFT.FQ.0.) GO TO 2	7	39	2		
1		DO 3 I=L,N	5	162			
3		A(I,I)=A(I,I)-SHIFT	18	252			
2		M=N-1	7	35			
		DO 4 T=L,M	7	162			
		DENOM= SORT(A(I,T)**2+A(I+1,I)**2)	18	3312			
		IF (DENOM.GT.0.) GO TO 10	18	126	18		
11		C(I)=1.0	***0*				
		S(I)=0.0	***0*				
		GO TO 4	***0*				
10		C(I)=A(I,I)/DENOM	18	288			
		S(I)=A(I+1,I)/DENOM	18	342			
		A(I,I)=DENOM	18	126			
12		I1=I+1	18	90			
		DO 35 J=I1,N	18	297			
		F=A(I,J)	33	231			
		A(I,J)=A(I+1,J)*S(I)+F*C(I)	33	957			
		A(I+1,J)=A(I+1,J)*C(I)-F*S(I)	33	1056			
35		CONTINUE	33				
4		CONTINUE	18				
C		TRIANGULAR MATRIX FORMED					
		DO 5 J=L,M	7	162			
		DO 6 I=L,J	18	297			
		F=A(I,J)	33	231			
		A(I,J)=A(I,J+1)*S(J)+F*C(J)	33	957			
6		A(I,J+1)=A(I,J+1)*C(J)-F*S(J)	33	1056			
		A(J+1,J)=A(J+1,J+1)*S(J)	18	450			
5		A(J+1,J+1)=A(J+1,J+1)*C(J)	18	504			
C		TRANSFORMATION COMPLETED					
		IF (SHIFT.FQ.0.) RETURN	7	39	2		
7		DO 9 I=L,N	5	162			
9		A(I,I)=A(I,I)+SHIFT	18	252			
8		RETURN	5	10			
		END					

	STATEMENT	EXECUTIONS	TIME	TRUE	
	SUBROUTINE COMPT(A,NMAX,L,N,RHO,SIGMA)	**0*			
	DIMENSION A(NMAX,NMAX)				
	C HOUSEHOLDER TYPE MATRICES USED INSTEAD OF ROTATIONS(SEE FRANCIS PAPER)				
	M=N-1	**0*			
	DO 1 I=L,M	**0*			
	IF (I.NE.L) GO TO 3	**0*			
	2 R1=A(L,L)*A(L,L)-SIGMA+A(L,L+1)*A(L+1,L)+RHO	**0*			
	R2=A(L+1,L)*A(L,L)+A(L+1,L+1)-SIGMA	**0*			
	B3=A(L+1,L)*A(L+2,L+1)	**0*			
	A(L+2,L)*0.0	**0*			
	GO TO 6	**0*			
	3 B1=A(L,I-1)	**0*			
	R2=A(I+1,I-1)	**0*			
	IF (I.EQ.M) GO TO 5	**0*			
	4 R3=A(I+2,I-1)	**0*			
	GO TO 6	**0*			
	5 R3=0.0	**0*			
	6 D=SQRT(B1*B1+R2*R2+R3*R3)	**0*			
	IF (R1.GF.D) GO TO 8	**0*			
	7 D=-D	**0*			
	8 IF (D.FQ.D) GO TO 1	**0*			
	C1=B2/(D+B1)	**0*			
	C2=B3/(D+R1)	**0*			
	F=2.0/(1.0+C1*C1+C2*C2)	**0*			
	DO 10 K=I,N	**0*			
	IF (I.EQ.M) GO TO 12	**0*			
	11 G=F*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))	**0*			
	A(I,K)=A(I,K)-G	**0*			
	A(I+1,K)=A(I+1,K)-C1*G	**0*			
	A(I+2,K)=A(I+2,K)-C2*G	**0*			
	GO TO 10	**0*			
	12 G=F*(A(I,K)+C1*A(I+1,K))	**0*			
	A(I,K)=A(I,K)-G	**0*			
	A(I+1,K)=A(I+1,K)-C1*G	**0*			
	10 CONTINUE	**0*			
	IF (I.EQ.L) GO TO 14	**0*			
	13 A(I,I-1)=D	**0*			
	A(I+1,I-1)=0.0	**0*			
	IF (I.EQ.M) GO TO 14	**0*			
	15 A(I+2,I-1)=0.0	**0*			
	C ROW OPERATION COMPLETED				
	14 J=I+2	**0*			
	IF (J.LE.N) GO TO 17	**0*			
	16 J=N	**0*			
	17 DO 18 K=L,J	**0*			
	IF (I.EQ.M) GO TO 20	**0*			
	19 G=F*(A(K,I)+C1*A(K,I+1)+C2*A(K,I+2))	**0*			
	A(K,I)=A(K,I)-G	**0*			
	A(K,I+1)=A(K,I+1)-C1*G	**0*			
	A(K,I+2)=A(K,I+2)-C2*G	**0*			
	GO TO 18	**0*			
	20 G=F*(A(K,I)+C1*A(K,I+1))	**0*			
	A(K,I)=A(K,I)-G	**0*			
	A(K,I+1)=A(K,I+1)-C1*G	**0*			
	18 CONTINUE	**0*			

STATE	TS	*** FORTUNE FOR IBM/360 ***	EXECUTIONS	TIME	TRUE	8
		J=I+3	**0*			
		IF (J.GT.N) GO TO 1	**0*			
21		G=F+C2*A(I+3,I+2)	**0*			
		A(I+3,I)=G	**0*			
		A(I+3,I+1)=C1*G	**0*			
		A(I+3,I+2)=A(I+3,I+2)+C2*G	**0*			
C		CO,UMN OPERATION COMPLETED				
1		CONTINUE	**0*			
		RETURN	**0*			
		END				

STATE	UNF	IBR	EXECUTIONS	TIME	TRUE	9
SUBROUTINE RVEC(A,NMAX,INT,C1,SMALL,N,NUH,EIGVEC,NVEC,INTER,Y)			4	216		
DIMENSION A(NMAX,NMAX),INTER(1),Y(1),INT(1),EIGVEC(NMAX,NMAX)						
DD1 I=1,N			4	144		
1 A(I,I)=A(I,I)+C1			16	224		
DD2 I=2,N			4	106		
IF(ABS(A(I-1,I-1)).GE.ABS(A(I,I-1)))GO TO 4			12	384	6	
3 INTER(I)=I			6	24		
J=I-1			6	30		
DD5 K=J,N			6	162		
B=A(I-1,K)			18	162		
A(I-1,K)=A(I,K)			18	252		
5 A(I,K)=B			18	126		
GO TO 6			6	12		
4 INTER(I)=I-1			6	42		
6 IF(A(I,I-1).EQ.0.) GO TO 2			12	144	**D*	
C=A(I,I-1)/A(I-1,I-1)			12	348		
A(I,I-1)=C			12	108		
DD8 K=I,N			12	216		
8 A(I,K)=A(I,K)+C*A(I-1,K)			24	600		
2 CONTINUE			12			
C TRIANGULAR MATRIX FORMED						
DD9 I=1,N			4	144		
IF(ABS(A(I,I)).GE.SMALL)GO TO 9			16	256	16	
10 A(I,I)=SMALL			**0*			
9 CONTINUE			16			
C NORMALISED ((EIGENVECTOR)) OF HESSENBURG MATRIX OBTAINED FOR RHS = 1						
DD 5555 I=1,N			4	144		
5555 Y(I)=1.0			16	64		
DD 8888 JJ,IJ=1,4			4	144		
7777 Y(N)=Y(N)/A(N,N)			16	288		
R=Y(N)			16	64		
DD11 I=2,N			16	432		
E=0.0			48	96		
M=N+1-I			48	336		
DD12 J=2,I			48	864		
L=M+J-I			96	672		
12 E=E+Y(L)*A(M,L)			96	1536		
Y(M)=(Y(M)-E)/A(M,M)			48	1008		
IF(ABS(B).GE.ABS(Y(M)))GO TO 11			48	856	20	
13 B=Y(M)			28	112		
11 CONTINUE			48			
DD14 I=1,N			16	576		
14 Y(I)=Y(I)/B			64	896		
IF(JJJJ.EQ.4)GO TO 6666			16	88	4	
DD15 I=2,N			12	324		
IF(INTER(I).NE.1) GO TO 15			36	288	18	
16 R=Y(I)			18	72		
Y(I)=Y(I-1)			18	162		
Y(I-1)=B			18	126		
15 Y(I)=Y(I)+A(I,I-1)*Y(I-1)			36	900		
8888 CONTINUE			12			
C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED (NOT NORMALIZED)						
6666 M=N-1			4	20		
DD21 I=2,M			4	72		
J=N+1-I			8	56		

STATE	IS	*** FORTUNE FOR IBM/360 ***	EXECUTIONS	TIME	TRUE	P	10
		L=J+1	8	40			
		DO20 K=L,N	8	108			
20		Y(K)=Y(K)-Y(J)*A(K,J=1)	12	264			
		LL=INT(N=1)	8	160			
		IF (J.EQ,LI) GO TO 21	8	48	4		
19		R=Y(J)	4	16			
		Y(J)=Y(LL)	4	24			
		Y(LL)=R	4	16			
21		CONTINUE	8				
		C UNNORMALIZED EIGENVECTOR OF ORIGINAL MATRIX OBTAINED					
		B=0.0	4	8			
		DO22 I=1,N	4	144			
		IF (ABS(B).GE. ABS(Y(I)))GO TO 22	16	288	8		
23		R=Y(I)	8	32			
22		CONTINUE	16				
		DO24 I=1,N	4	144			
24		Y(I)=Y(I)/R	16	224			
		C NORMALIZED EIGENVECTOR OBTAINED					
		C PUT Y(I) (REAL EIGENVECTOR) IN EIGVECT(NUMOFEV,I)					
		DO 100 I=1,N	4	144			
100		EIGVEC(NUM,I)=Y(I)	16	144			
		RETURN	4	8			
		END					

```
SUBROUTINE IHVFC(A,NMAX,INT,C1,AHU,SMALL,N,NUH,ETGVEC,NVEC,INTER,
1Y,7,R)
DIMENSION A(NMAX,NMAX),INTER(1),Y(1),Z(1),INT(1),B(NMAX,NMAX),
1EIRVFC(NMAX,NMAX)
DO40 I=1,N
DO50 J=1,N
50 B(I,J)=0.0
DO1 I=1,N
A(I,1)=A(I,1)-C1
1 B(I,1)=B(I,1)+AHU
DO2 I=2,N
C=A(I-1,I-1)+A(I-1,I)+B(I-1,I)+B(I-1,I-1)
D=A(I,I-1)+A(I,I)+B(I,I-1)+B(I,I-1)
IF (C.GE.D) GO TO 4
3 INTER(I)=I
G=C
C=D
D=G
J=Y-1
DO4 K=J,N
G=A(I-1,K)
A(I-1,K)=A(I,K)
A(I,K)=G
G=B(I-1,K)
B(I-1,K)=B(I,K)
5 B(I,K)=G
GO TO 6
4 INTER(I)=I-1
6 IF (D.FO.0) GO TO 2
7 G=A(I,I-1)
A(I,I-1)=(G*A(I-1,I-1)+B(I,I-1)+B(I-1,I-1))/C
B(I,I-1)=(B(I,I-1)*A(I-1,I-1)+G*B(I-1,I-1))/C
DO4 K=I,N
A(I,K)=A(I,K)+A(I,I-1)*A(I-1,K)-B(I,I-1)*B(I-1,K)
8 B(I,K)=B(I,K)+B(I,I-1)*A(I-1,K)+A(I,I-1)*B(I-1,K)
2 CONTINUE
C TRIANGULAR MATRIX FORMED
DO9 I=1,N
C=A(I,I)+A(I,I)+B(I,I)+B(I,I)
IF (C.GE.SMALL) GO TO 9
A(I,I)=SQRT(SMALL)
B(I,I)=0.0
9 CONTINUE
DO 1111 I=1,N
Y(I)=1.0
1111 Z(I)=0.0
DO 8888 JJ=1,4
C=A(N,N)+A(N,N)+B(N,N)+B(N,N)
G=Y(N)
Y(N)=(G*A(N,N)+Z(N)+B(N,N))/C
Z(N)=(Z(N)+A(N,N)+G*B(N,N))/C
DO17 I=2,N
E=0.0
F=0.0
H=N+1=I
```

ORIGINAL PAGE IS
OF POOR QUALITY

STATE	TS	*** FORTUNE FOR IBM/360 ***	EXECUTIONS	TIME	TRUE	12
		0018 J=2,I	**0*			
		L=N+J-I	**0*			
		E=F+Y(L)*A(M,L)-Z(L)*B(M,L)	**0*			
18		F=F+Y(L)*B(M,L)+Z(L)*A(M,L)	**0*			
		D=A(M,M)*A(M,M)+B(M,M)*B(M,M)	**0*			
		G=Y(M)	**0*			
		Y(M)=((G-E)*A(M,M)+(Z(M)=F)*B(M,M))/D	**0*			
17		Z(M)=((Z(M)=F)*A(M,M)+(G-E)*B(M,M))/D	**0*			
		E=0.0	**0*			
		DO51 I=1,N	**0*			
		F=Y(I)*Y(I)+Z(I)*Z(I)	**0*			
		IF (E,GE,F) GO TO 51	**0*			
52		F=F	**0*			
		L=I	**0*			
51		CONTINUE	**0*			
		A1=Y(L)	**0*			
		A2=Z(L)	**0*			
		DO14 I=1,N	**0*			
		G=Y(I)	**0*			
		Y(I)=(G*A1+Z(I)*A2)/E	**0*			
14		Z(I)=(Z(I)*A1+G*A2)/E	**0*			
		IF (JJJ,EQ,4) GO TO 6666	**0*			
		DO15 I=2,N	**0*			
		IF (INTER(I),NE,I) GO TO 30	**0*			
16		G=Y(I)	**0*			
		Y(I)=Y(I-1)	**0*			
		Y(I-1)=G	**0*			
		G=7(I)	**0*			
		Z(I)=Z(I-1)	**0*			
		Z(I-1)=G	**0*			
30		Y(I)=Y(I)+A(I,I-1)*Y(I-1)-R(I,I-1)*Z(I-1)	**0*			
15		Z(I)=Z(I)+R(I,I-1)*Y(I-1)+A(I,I-1)*Z(I-1)	**0*			
8888		CONTINUE	**0*			
C		EIGENVECTOR OF HESSENBURG MATRIX OBTAINED				
6666		M*N=1	**0*			
		DO21 I=2,M	**0*			
		J=N+1-I	**0*			
		L=I+1	**0*			
		DO20 K=L,N	**0*			
		Y(K)=Y(K)-Y(J)*A(K,J-1)+Z(J)*B(K,J-1)	**0*			
20		Z(K)=Z(K)-7(J)*A(K,J-1)+Y(J)*B(K,J-1)	**0*			
		LL=INT(M-I)	**0*			
		IF (J,EQ,LL) GO TO 21	**0*			
19		G=Y(J)	**0*			
		Y(J)=Y(LL)	**0*			
		Y(LL)=G	**0*			
		G=7(J)	**0*			
		Z(J)=Z(LL)	**0*			
		Z(LL)=G	**0*			
21		CONTINUE	**0*			
C		UNNORMALIZED EIGENVECTOR OBTAINED				
		E=0.0	**0*			
		DO22 I=1,N	**0*			
		F=Y(I)*Y(I)+Z(I)*Z(I)	**0*			
		IF (E,GE,F) GO TO 22	**0*			

```

23 F=F
L=L
22 CONTINUE
A1=Y(L)
A2=Z(L)
DO 24 I=1,N
G=Y(I)
Y(I)=(G*A1+Z(I)*A2)/E
24 Z(I)=(Z(I)+A1-G*A2)/E
C NORMALIZED EIGENVECTOR OBTAINED
C PUT Y(I) (REAL PART) IN EIGVECT(NUMFEV,I)
DO 100 I=1,N
100 EIGVEC(NUM,I)=Y(I)
NUM=NUM+1
C PUT Z(I) (COMPLEX PART) IN EIGVECT(NUMFEV,I)
DO 101 I=1,N
101 EIGVEC(NUM,I)=Z(I)
RETURN
END

```

*** PROGRAM SUMMARY -

ROUTINE	TIME	PERCENT
MAIN	8054	22.3
CALCEV	485	1.3
REALIT	11903	32.9
COMPTT	0	0.0
RVEC	15710	43.5
INVEC	0	0.0

	36152	

FORTRAN FREQUENCIES: OCT73: N1.

C FOR DESCRIPTION OF METHOD USED IN THIS PROGRAM, SEE **THE QR
C TRANSFORMATION** BY FRANCIS, COMPUTER JOURNAL, VOL. 4 PP. 265,332
C START HERE TO PRODUCE UPPER HESSIAN MATRIX
C DIMENSION A(10,10),VEC(3), EIGVEC(4,4), INT(10),CC(10),SS(10)
C DIMENSION Z(10),D(10,10),BR(10,10)
C DATA A / 0.1,5.8,1.9,3.2,6*0.0,2.8,3.9,0.7,0.1,6*0.0,
C Z 6.2,3.5,0.6,1.2,6*0.0,9.9,6.2,1.7,3.8,66*0.0/
C ***** SEGMENT 1 *****
C N=10
C M=4
C NVEC=4
C N=N-2
C K=N-2
C DO1J=1,K
C ***** SEGMENT 2 *****
C K1=J+1
C K2=J+2
C GREAT=ABS(A(K1,J))
C MAX=K1
C DO2I=K2,N
C ***** SEGMENT 3 *****
C A1=ABS(A(I,J))
C ***** SEGMENTS 4 THRU 5 *****
C IF (GREAT.GE.A1) GO TO 2
C ***** SEGMENT 6 *****
C 3 GREAT=A1
C MAX=I
C ***** SEGMENT 7 *****
C 2 CONTINUE
C INT(I) GIVES INFORMATION ABOUT INTERCHANGES
C ***** SEGMENT 8 *****
C INT(J)=MAX
C ***** SEGMENTS 9 THRU 10 *****
C IF (MAX.EQ.K1) GO TO 5
C ***** SEGMENT 11 *****
C 4 DO6L=J,N
C ***** SEGMENT 12 *****
C F=A(K1,L)
C A(K1,L)=A(MAX,L)
C ***** SEGMENT 13 *****
C 6 A(MAX,L)=F
C ***** SEGMENT 14 *****
C DO7L=L,N
C ***** SEGMENT 15 *****
C F=A(L,K1)
C A(L,K1)=A(L,MAX)
C ***** SEGMENT 16 *****
C 7 A(L,MAX)=F
C INTERCHANGE DONE
C ***** SEGMENT 17 *****
C 5 DO9I=K2,N
C ***** SEGMENTS 18 THRU 19 *****
C IF (A(I,J).EQ.0.) GO TO 4
C ***** SEGMENT 20 *****

ORIGINAL PAGE IS
OF POOR QUALITY

1C

```

      A(I,J)=-A(I,J)/A(K1,J)
      DO11M=K1,N
C*****SEGMENT      21 *****
      11 A(I,M)=A(I,M)+A(I,J)*A(K1,M)
C*****SEGMENT      22 *****
      8 CONTINUE
      C LEFT MULT. DONE
C*****SEGMENT      23 *****
      DO 2000 M=K2,N
C*****SEGMENTS 24 THRU 25 *****
      IF (A(M,J).EQ.0.) GO TO 2000
C*****SEGMENT      26 *****
      10 DO 2001 I=1,M
C*****SEGMENT      27 *****
      2001 A(I,K1)=A(I,K1)-A(I,M)*A(M,J)
C*****SEGMENT      28 *****
      3000 CONTINUE
C*****SEGMENT      29 *****
      1 CONTINUE
      C RIGHT MULT. DONE
      C HESSBURG MATRIX NOW FORMED AS A(I,J). MULTIPLIERS STORED WITH IT
C*****SEGMENT      30 *****
      DO 105 I=1,N
C*****SEGMENT      31 *****
      DO 105 J=1,N
C*****SEGMENT      32 *****
      105 A(I,J)=A(I,J)
      C PUT APPROPRIATE ZEROS WHERE MULTIPLIERS WERE STORED
C*****SEGMENT      33 *****
      DO 106 I=3,N
C*****SEGMENT      34 *****
      L=I-2
      DO 106 J=1,L
C*****SEGMENT      35 *****
      106 A(I,J)=0.0
C*****SEGMENT      36 *****
      K1=M+1
      R=0.0
      C=0.0
      DO33I=1,N
C*****SEGMENT      37 *****
      R=R+ABS(A(I,I))
C*****SEGMENT      38 *****
      33 C=C+ABS(A(I,N))
C*****SEGMENT      39 *****
      DO34I=2,N
C*****SEGMENT      40 *****
      R=R+0.0
      C=C+0.0
      L=I-1
      DO35J=L,N
C*****SEGMENT      41 *****
      R=R+ABS(A(I,J))
      K1=K1-J
      K1I=K1-I
C*****SEGMENT      42 *****
      35 C=C+ABS(A(K1J,K1I))
C*****SEGMENTS 43 THRU 44 *****

```

```

      IF (MS.LE.R) GO TO 36
C***** SEGMENT 45 *****
      37 PERS
C***** SEGMENTS 46 THRU 47 *****
      38 IF (CS.LE.C) GO TO 34
C***** SEGMENT 48 *****
      44 C=CS
C***** SEGMENT 49 *****
      34 CONTINUE
C***** SEGMENT 50 *****
      IJK=1
C***** SEGMENTS 51 THRU 52 *****
      IF (P.GL.C) GO TO 54
C DELTA IS CRITERION FOR STOPPING AN ITERATION
C***** SEGMENT 53 *****
      53 DELTA=R*1.0E-11
C DEL IS ARBITRARY SHIFT
      DEL=P/17.1829
      GO TO 153
C***** SEGMENT 54 *****
      54 DELTA=C+1.0E-11
      DEL=C/17.1829
C DELTA COMPUTED
C PRODUCE (-P), ITERATE ON THIS TO BREAK UP EIGENVALUES OF EQUAL MODULUS
C***** SEGMENT 55 *****
      153 DO 107 I=1,N
C***** SEGMENT 56 *****
      107 A(I,I)=A(I,I)-DEL
C***** SEGMENTS 57 THRU 58 *****
      55 IF (P.GL.1) GO TO 48
C SUBROUTINE CALCEV CALCULATES EIGENVALUES OF 1X1 OR 2X2 MATRICES.
C P1 IS ADDED BACK IN CALCEV
C***** SEGMENT 59 *****
      47 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
      GO TO 76
C***** SEGMENTS 60 THRU 61 *****
      48 IF (N.NE.2) GO TO 80
C***** SEGMENT 62 *****
      49 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
      GO TO 76
C OR TRANSFORMATION STARTS HERE
C INITIALIZE FOR DETERMINING SUBSEQUENT SHIFTS
C***** SEGMENT 63 *****
      80 AR1=0.0
      AR3=0.0
      AR4=0.0
C RE-ENTER AFTER REALIT OR COMPT SURROUTINES CALLED
C SELECT FIRST ((ZERO)) SUB-DIAGONAL ELEMENT FROM BOTTOM
C***** SEGMENT 64 *****
      61 L=N
      D056 I=2,N
C***** SEGMENT 65 *****
      N2I=N+2-I
C***** SEGMENTS 66 THRU 67 *****
      IF (ABS(A(N2I,N2I-1)).LT.DELTA) GO TO 57
C***** SEGMENT 68 *****
      56 L=L-1
C VALUE OF L SELECTED

```

ORIGINAL PAGE IS
 OF POOR QUALITY

36

```

C***** SEGMENTS 69 THRU 70 *****
57 IF ((L.NE.N) GO TO 59
C***** SEGMENT 71 *****
58 CALL CALCEV(A,NMAX,DFL,VEC,IJK,1)
N=N-1
GO TO 48
C***** SEGMENTS 72 THRU 73 *****
59 IF ((L=N+1).NE.D) GO TO 75
C***** SEGMENT 74 *****
60 CALL CALCEV(A,NMAX,DFL,VEC,IJK,2)
N=N-2
GO TO 55
C DETERMINE EIGENVALUES OF 2X2 MATRIX -- ROW 1 = A(N-1,N-1)  A(N-1,N)
C ROW 2 = A(N,N-1)  A(N,N)
C AND BASE SHIFT ON THESE
C***** SEGMENT 75 *****
75 H=A(N-1,N-1)+A(N,N)
C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
C1=A(N-1,N-1)-A(N,N)
DISC=C1+C1+4.0*A(N,N-1)*A(N-1,N)
C***** SEGMENTS 76 THRU 77 *****
IF (DISC.GE.0.) GO TO 83
C***** SEGMENT 78 *****
82 R1=0.5*H
R2=.5*SQRT(ABS(DISC))
NUM=(R1-AR3)*((R1-AR3)+(R2-AR4))
DENOM=AR3*AR3+AR4*AR4
C***** SEGMENTS 79 THRU 80 *****
IF (DENOM.NE.0.) GO TO 200
C***** SEGMENT 81 *****
201 RHO=0.0
SIGMA=0.0
GO TO 203
C***** SEGMENT 82 *****
200 RHO=0
SIGMA=0
C***** SEGMENT 83 *****
203 AR3=1
AR4=2
CALL COMPT(A,NMAX,L,N,RHO,SIGMA)
GO TO 61
C***** SEGMENTS 84 THRU 85 *****
83 IF (R.GE.0.) GO TO 85
C***** SEGMENT 86 *****
84 R1=.5*(H+SQRT(DISC))
GO TO 86
C***** SEGMENT 87 *****
85 R1=.5*(H+SQRT(DISC))
C***** SEGMENTS 88 THRU 89 *****
IF (R1.NE.0.) GO TO 86
C***** SEGMENT 90 *****
83 R2=0.0
GO TO 84
C***** SEGMENT 91 *****
86 R2=C/R1
C ROOTS OF 2*2 MATRIX FOUND
C***** SEGMENTS 92 THRU 93 *****
84 IF (ABS(A(N,N)-R1).GE.0.5*ABS(A(N,N)-R2)) GO TO 88

```

4C

```

C***** SEGMENT 94 *****
  67 AR2=P1
  68 AR2=P2
C***** SEGMENTS 94 THRU 97 *****
  69 IF (AR1.NE.0.) GO TO 71
C***** SEGMENT 98 *****
  72 SHIFT=0.0
  69 TO 73
C***** SEGMENT 99 *****
  71 SHIFT=AR2
C***** SEGMENT 100 *****
  73 AR1=AR2
  CALL REWIT(A,NMAX,L,N,SHIFT,CC,SS)
  69 TO 61
C ARE VECTORS NEEDED
C***** SEGMENTS 101 THRU 102 *****
  76 IF (NVEC.EQ.0) GO TO 600
C INVERSE ITERATION IS USED FOR EIGENVECTORS
C***** SEGMENT 103 *****
  550 SMALL=DLTA*1.0E-30
  NUM=0
  JI=1
  NN=NN+NN
C C1 IS REAL PART OF EIGENVALUE
C***** SEGMENT 104 *****
  513 C1=VFC(JI)
C C2 IS COMPLY PART OF EIGENVALUE
  C2=VFC(JI+1)
  NUM=NUM+1
C REPLACE HESSENBURG MATRIX AND MULTIPLIERS
  60 570 I=1,NN
C***** SEGMENT 105 *****
  60 570 J=1,NN
C***** SEGMENT 106 *****
  570 A(I,J)=A(I,J)
C***** SEGMENTS 107 THRU 108 *****
  IF (C2.NE.0.) GO TO 561
C***** SEGMENT 109 *****
  CALL RVLC(A,NMAX,INT,C1,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS)
  JI=JI+2
C***** SEGMENTS 110 THRU 111 *****
  IF (NVEC.EQ.NN) GO TO 600
C***** SEGMENT 112 *****
  60 TO 513
C***** SEGMENT 113 *****
  561 CALL INVFC(A,NMAX,INT,C1,C2,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS,2,NN)
C NEXT STATEMENT IGNORES CONJUGATE COMPLEX EIGENVALUES
  JI=JI+4
C***** SEGMENTS 114 THRU 115 *****
  IF (NVEC.EQ.NN) GO TO 600
C***** SEGMENT 116 *****
  60 TO 513
C***** SEGMENT 117 *****
  600 CONTINUE
  AR1=(A(700)) (VEC(I),I=1,NN)
  60 700 I=1,NN

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

C***** SEGMENT 118 *****
20 WRITE(6,700) (EIGVEC(I,J),J=1,NH)
700 FORMAT(1H,5F20.8 / 5F20.8)
END

```

END PASS1, VALUE= 193

```

SUBROUTINE CALCEV(A,NMAX,DFL,VEC,IJK,N,M)
DIMENSION A(NMAX,NMAX),VEC(1)

```

```

C***** SEGMENTS 119 THRU 120 *****
IF (N.NE.1) GO TO 2

```

```

C***** SEGMENT 121 *****
1 A1=0.0

```

```

A(I,I)=A(I,I)+DEL

```

```

C STORE EIGENVALUES IN VEC

```

```

VEC(IJK)=A(I,I)

```

```

VEC(IJK+1)=A1

```

```

IJK=IJK+2

```

```

RETURN

```

```

C***** SEGMENT 122 *****
2 F=A(I-1,I-1)+A(I,I,N)

```

```

C=A(I-1,I,N)+A(I,I,N)-A(N,I-1)*A(N-1,N)

```

```

C1=A(I-1,I-1)-A(I,I,N)

```

```

DISC=C1+C1+4.0*A(N,I-1)*A(N-1,N)

```

```

C***** SEGMENTS 123 THRU 124 *****
IF (ISC.LT.0.) GO TO 9

```

```

C***** SEGMENT 125 *****
4 A1=0.0

```

```

C***** SEGMENTS 126 THRU 127 *****
IF (N.GE.0.) GO TO 6

```

```

C***** SEGMENT 128 *****
R1=0.5*(N-SORT(DISC))

```

```

GO TO 7

```

```

C***** SEGMENT 129 *****
6 R1=0.5*(R1-SORT(DISC))

```

```

C***** SEGMENTS 130 THRU 131 *****
IF (N1.NE.0.) GO TO 7

```

```

C***** SEGMENT 132 *****
10 R2=0.0

```

```

GO TO 11

```

```

C***** SEGMENT 133 *****
7 R2=C/R1

```

```

C***** SEGMENT 134 *****
11 R1=R1+DEL

```

```

R2=R2+DEL

```

```

VEC(IJK)=R2

```

```

VEC(IJK+1)=A1

```

```

VEC(IJK+2)=R1

```

```

VEC(IJK+3)=A1

```

```

GO TO 8

```

```

C***** SEGMENT 135 *****
9 R1=0.5*0

```

```

A1=-0.5*SORT(ABS(DISC))

```

```

R1=R1+DEL

```

```

VEC(IJK)=R1

```

```

VEC(IJK+1)=-A1

```

```

VEC(IJK+2)=R1

```

```

VEC(IJK+3)=A1

```

```

GO TO 8

```

```

C***** SEGMENT 136 *****

```

60

```

      B IJK=IJK+4
      RETURN
      END

```

```

END PASS1, VALUE= 41
SUBROUTINE REALIT(A,NMAX,L,M,SHIFT,C,S)
  DIMENSION A(NMAX,NMAX),C(1),S(1)
C***** SEGMENTS 137 THRU 138 *****
  IF (SHIFT.EQ.0.) GO TO 2
C***** SEGMENT 139 *****
  1 DO3 I=L,M
C***** SEGMENT 140 *****
  3 A(I,I)=A(I,I)-SHIFT
C***** SEGMENT 141 *****
  2 I=M-1
  DO4 I=L,M
C***** SEGMENT 142 *****
  DENOM= SQRT(A(I,I)**2+A(I+1,I)**2)
C***** SEGMENTS 143 THRU 144 *****
  IF (DENOM.GT.0.) GO TO 10
C***** SEGMENT 145 *****
  11 C(I)=1.0
  S(I)=0.0
  GO TO 4
C***** SEGMENT 146 *****
  10 C(I)=A(I,I)/DENOM
  S(I)=A(I+1,I)/DENOM
  A(I,I)=DENOM
C***** SEGMENT 147 *****
  12 I=I+1
  DO 35 J=I+1,M
C***** SEGMENT 148 *****
  F=A(I,J)
  A(I,J)=A(I+1,J)*S(I)+F*C(I)
  A(I+1,J)=A(I+1,J)*C(I)-F*S(I)
C***** SEGMENT 149 *****
  35 CONTINUE
C***** SEGMENT 150 *****
  4 CONTINUE
C TRIANGULAR MATRIX FORMED
C***** SEGMENT 151 *****
  DO5 I=L,M
C***** SEGMENT 152 *****
  DO6 I=L,J
C***** SEGMENT 153 *****
  F=A(I,J)
  A(I,J)=A(I,J+1)*S(J)+F*C(J)
C***** SEGMENT 154 *****
  6 A(I,J+1)=A(I,J+1)*C(J)-F*S(J)
C***** SEGMENT 155 *****
  A(J+1,J)=A(J+1,J+1)*S(J)
C***** SEGMENT 156 *****
  5 A(J+1,J+1)=A(J+1,J+1)*C(J)
C TRANSFORMATION COMPLETED
C***** SEGMENTS 157 THRU 158 *****
  IF (SHIFT.EQ.0.) RETURN
C***** SEGMENT 159 *****
  7 DO9 I=L,M

```

ORIGINAL PAGE IS
 OF POOR QUALITY

7c

```

C+***** SEGMENT 160 *****
9 A(I,I)=A(I,I)+SHIFT
C+***** SEGMENT 161 *****
R RETURN
END

END PASS1, VALUE= 36
SUBROUTINE COMPIT(A,NMAX,L,N,RHO,SIGMA)
DIMENSION A(NMAX,NMAX)
C HOUSEHOLDER TYPE MATRICES USED INSTEAD OF ROTATIONS(SFE FRANCIS PAPER)
C+***** SEGMENT 162 *****
MEN=1
DO 1 I=L,M
C+***** SEGMENTS 163 THRU 164 *****
IF (I.NE.L) GO TO 3
C+***** SEGMENT 165 *****
2 B1=A(L,L)*(A(L,L)-SIGMA)+A(L,L+1)*A(L+1,L)+RHO
B2=A(L+1,L)*(A(L,L)+A(L+1,L+1)-SIGMA)
B3=A(L+1,L)+A(L+2,L+1)
A(L+2,L)=0.0
GO TO 6
C+***** SEGMENT 166 *****
3 B1=A(I,I-1)
B2=A(I+1,I-1)
C+***** SEGMENTS 167 THRU 168 *****
IF (I.EQ.M) GO TO 5
C+***** SEGMENT 169 *****
4 B3=A(I+2,I-1)
GO TO 6
C+***** SEGMENT 170 *****
5 B3=0.0
C+***** SEGMENT 171 *****
6 D=SQRT(B1*B1+B2*B2+B3*B3)
C+***** SEGMENTS 172 THRU 173 *****
IF (D.EQ.0.) GO TO 8
C+***** SEGMENT 174 *****
7 D=-D
C+***** SEGMENTS 175 THRU 176 *****
8 IF (D.EQ.0.) GO TO 1
C+***** SEGMENT 177 *****
9 C1=B2/(D+B1)
C2=B3/(D+B1)
F=2.0/(1.0+C1*C1+C2*C2)
DO 10 K=I,N
C+***** SEGMENTS 178 THRU 179 *****
IF (I.EQ.K) GO TO 12
C+***** SEGMENT 180 *****
11 G=F*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))
A(I,K)=A(I,K)-G
A(I+1,K)=A(I+1,K)-C1*G
A(I+2,K)=A(I+2,K)-C2*G
GO TO 10
C+***** SEGMENT 181 *****
12 G=F*(A(I,K)+C1*A(I+1,K))
A(I,K)=A(I,K)-G
A(I+1,K)=A(I+1,K)-C1*G
C+***** SEGMENT 182 *****
10 CONTINUE

```



```

C***** SEGMENTS 183 THRU 184 *****
IF (I, EQ, L) GO TO 14
C***** SEGMENT 185 *****
13 A(I, I-1) = -D
A(I+1, I-1) = 0.0
C***** SEGMENTS 186 THRU 187 *****
IF (I, EQ, M) GO TO 14
C***** SEGMENT 188 *****
15 A(I+2, I-1) = 0.0
C ROW OPERATION COMPLETED
C***** SEGMENT 189 *****
14 J = I+2
C***** SEGMENTS 190 THRU 191 *****
IF (J, LE, N) GO TO 17
C***** SEGMENT 192 *****
16 J = N
C***** SEGMENT 193 *****
17 DO 18 K = I, J
C***** SEGMENTS 194 THRU 195 *****
IF (I, EQ, K) GO TO 20
C***** SEGMENT 196 *****
19 G = F*(A(K, I)+C1*A(K, I+1)+C2*A(K, I+2))
A(K, I) = A(K, I) - G
A(K, I+1) = A(K, I+1) - C1*G
A(K, I+2) = A(K, I+2) - C2*G
GO TO 18
C***** SEGMENT 197 *****
20 G = F*(A(K, I)+C1*A(K, I+1))
A(K, I) = A(K, I) - G
A(K, I+1) = A(K, I+1) - C1*G
C***** SEGMENT 198 *****
18 CONTINUE
C***** SEGMENT 199 *****
J = I+3
C***** SEGMENTS 200 THRU 201 *****
IF (J, GT, N) GO TO 1
C***** SEGMENT 202 *****
21 G = F*C2*A(I+3, I+2)
A(I+3, I) = -G
A(I+3, I+1) = -C1*G
A(I+3, I+2) = A(I+3, I+2) - C2*G
C COLUMN OPERATION COMPLETED
C***** SEGMENT 203 *****
1 CONTINUE
C***** SEGMENT 204 *****
RETURN
END

END PASG1, VALUE= 65
SUBROUTINE RVEC(A, NMAX, I, T, C1, SMALL, N, NUM, EIGVEC, NVEC, INTER, Y)
DIMENSION A(NMAX, NMAX), INTER(1), Y(1), INT(1), FIVEVEC(NMAX, NMAX)
C***** SEGMENT 205 *****
DO1 I=1, N
C***** SEGMENT 206 *****
1 A(I, I) = A(I, I) - C1
C***** SEGMENT 207 *****
DO2 I=2, N
C***** SEGMENTS 208 THRU 209 *****

```

```

      IF( ABS(A(I-1,I-1)).GE. ABS(A(I,I-1)))GO TO 4
C***** SEGMENT 210 *****
      3 INTER(I)=I
      J=I-1
      DO5 K=J,N
C***** SEGMENT 211 *****
      U=A(I-1,K)
      A(I-1,K)=A(I,K)
C***** SEGMENT 212 *****
      5 A(I,K)=U
C***** SEGMENT 213 *****
      GO TO 6
C***** SEGMENT 214 *****
      4 INTER(I)=I-1
C***** SEGMENTS 215 THRU 216 *****
      6 IF (A(I,I-1).EQ.0.) GO TO 2
C***** SEGMENT 217 *****
      C=-A(I,I-1)/A(I-1,I-1)
      A(I,I-1)=C
      DO 8 K=I,N
C***** SEGMENT 218 *****
      8 A(I,K)=A(I,K)+C*A(I-1,K)
C***** SEGMENT 219 *****
      2 CONTINUE
C      TRIANGULAR MATRIX FORMED
C***** SEGMENT 220 *****
      DO9 I=1,N
C***** SEGMENTS 221 THRU 222 *****
      IF( ABS(A(I,I)).GE.SMALL)GO TO 9
C***** SEGMENT 223 *****
      10 A(I,I)=SMALL
C***** SEGMENT 224 *****
      9 CONTINUE
C      NORMALISED ((EIGENVECTOR)) OF HESSENBURG MATRIX OBTAINED FOR PHS = 1
C***** SEGMENT 225 *****
      DO 5555 I=1,N
C***** SEGMENT 226 *****
      5555 Y(I)=1.0
C***** SEGMENT 227 *****
      DO 7777 J=1,N
C***** SEGMENT 228 *****
      7777 Y(J)=Y(J)/A(I,J)
      M=Y(M)
      DO11 I=2,N
C***** SEGMENT 229 *****
      F=0.0
      M=M+1-I
      DO12 J=2,I
C***** SEGMENT 230 *****
      L=M+J-I
C***** SEGMENT 231 *****
      12 F=F+Y(L)*A(M,L)
C***** SEGMENT 232 *****
      Y(M)=(Y(M)-F)/A(M,M)
C***** SEGMENTS 233 THRU 234 *****
      IF( ABS(F).GE. ABS(Y(M)))GO TO 11
C***** SEGMENT 235 *****
      13 M=Y(M)

```

100

```

C***** SEGMENT 236 *****
11 CONTINUE
C***** SEGMENT 237 *****
DO14 I=1,N
C***** SEGMENT 238 *****
14 Y(I)=Y(I)/B
C***** SEGMENTS 239 THRU 240 *****
IF (JJJJ.FJ.4) GO TO 6666
C***** SEGMENT 241 *****
DO15 I=2,N
C***** SEGMENTS 242 THRU 243 *****
IF (INTER(I).NE.I) GO TO 15
C***** SEGMENT 244 *****
16 H=Y(I)
Y(I)=Y(I-1)
Y(I-1)=H
C***** SEGMENT 245 *****
15 Y(I)=Y(I)+A(I,I-1)*Y(I-1)
C***** SEGMENT 246 *****
6666 CONTINUE
C EIGENVECTOR OF HESSENBERG MATRIX OBTAINED (NOT NORMALIZED)
C***** SEGMENT 247 *****
DO20 I=1,N
DO21 I=2,N
C***** SEGMENT 248 *****
J=I+1-I
L=J+1
DO20 K=L,N
C***** SEGMENT 249 *****
20 Y(K)=Y(K)-Y(J)*A(K,J-1)
C***** SEGMENT 250 *****
LL=INT(N-1)
C***** SEGMENTS 251 THRU 252 *****
IF (J.EQ.LL) GO TO 21
C***** SEGMENT 253 *****
19 B=Y(J)
Y(J)=Y(LL)
Y(LL)=B
C***** SEGMENT 254 *****
21 CONTINUE
C UNNORMALIZED EIGENVECTOR OF ORIGINAL MATRIX OBTAINED
C***** SEGMENT 255 *****
DO22 I=1,N
C***** SEGMENTS 256 THRU 257 *****
IF (ABS(H).GE. ABS(Y(I))) GO TO 22
C***** SEGMENT 258 *****
23 H=Y(I)
C***** SEGMENT 259 *****
22 CONTINUE
C***** SEGMENT 260 *****
DO23 I=1,N
C***** SEGMENT 261 *****
24 Y(I)=Y(I)/H
C UNNORMALIZED EIGENVECTOR OBTAINED
C PUT Y(I) (REAL EIGENVECTOR) IN FIVECT(UNOFFV,I)
C***** SEGMENT 262 *****
DO 100 I=1,N

```

ORIGINAL PAGE IS
OF POOR QUALITY

11C

```

C***** SEGMENT 263 *****
100 EIGVEC(NUM,I)=Y(I)
C***** SEGMENT 264 *****
RETURN
END

END PASS1, VALUE= 78
SUBROUTINE IMVEC(A,NMAX,INT,C1,AMU,SMALL,N,NUM,EIGVEC,NVEC,INTER,
1Y,Z,P)
DIMENSION A(NMAX,NMAX),INTER(1),Y(1),Z(1),INT(1),R(NMAX,NMAX),
1EIGVEC(NMAX,NMAX)
C***** SEGMENT 265 *****
DO50 I=1,N
C***** SEGMENT 266 *****
DO60 J=1,N
C***** SEGMENT 267 *****
50 R(I,J)=0.0
C***** SEGMENT 268 *****
DO1 I=1,N
C***** SEGMENT 269 *****
A(I,I)=A(I,I)-C1
C***** SEGMENT 270 *****
1 R(I,I)=B(I,I)-AMU
C***** SEGMENT 271 *****
DO2 I=2,N
C***** SEGMENT 272 *****
C=A(I-1,I-1)*A(I-1,I-1)+B(I-1,I-1)*B(I-1,I-1)
D=A(I,I-1)*A(I,I-1)+B(I,I-1)*B(I,I-1)
C***** SEGMENTS 273 THRU 274 *****
IF (C,GE,D) GO TO 4
C***** SEGMENT 275 *****
3 INTER(I)=I
DEC
C=D
DEC
J=I-1
DO5 K=J,N
C***** SEGMENT 276 *****
G=A(I-1,K)
A(I-1,K)=A(I,K)
A(I,K)=G
G=B(I-1,K)
B(I-1,K)=B(I,K)
C***** SEGMENT 277 *****
5 B(I,K)=G
C***** SEGMENT 278 *****
GO TO 6
C***** SEGMENT 279 *****
4 INTER(I)=I-1
C***** SEGMENTS 280 THRU 281 *****
6 IF (C,FG,D) GO TO 2
C***** SEGMENT 282 *****
7 C=A(I,I-1)
A(I,I-1)=-(G*A(I-1,I-1)+B(I,I-1)*B(I-1,I-1))/C
B(I,I-1)=-(B(I,I-1)+A(I-1,I-1)-G*B(I-1,I-1))/C
DO6 K=1,N
C***** SEGMENT 283 *****
A(I,K)=A(I,K)+A(I,I-1)*A(I-1,K)-B(I,I-1)*B(I-1,K)

```

```

C ***** SEGMENT 284 *****
  B(I,K)=B(I,K)+B(I,I-1)*A(I-1,K)+A(I,I-1)*R(I-1,K)
C ***** SEGMENT 285 *****
  2 CONTINUE
C ***** SEGMENT 286 *****
  TRIN*GULAR MATRIX FORMED
  DO9 I=1,N
C ***** SEGMENT 287 *****
  C=A(I,I)+A(I,I)+R(I,I)+B(I,I)
C ***** SEGMENTS 288 THRU 289 *****
  IF (C.GE.SMALL) GO TO 9
C ***** SEGMENT 290 *****
  A(I,I)=SQRT(SMALL)
  B(I,I)=,0
C ***** SEGMENT 291 *****
  9 CONTINUE
C ***** SEGMENT 292 *****
  DO 1111 I=1,N
C ***** SEGMENT 293 *****
  Y(I)=1.0
C ***** SEGMENT 294 *****
  1111 Z(I)=0.0
C ***** SEGMENT 295 *****
  DO 3389 JJJ=1,4
C ***** SEGMENT 296 *****
  C=A(I,JJJ)+A(N,JJJ)+B(N,JJJ)*B(I,JJJ)
  G=Y(I)
  Y(I)=(G+A(N,JJJ)+Z(JJJ)+B(N,JJJ))/C
  Z(JJJ)=(Z(JJJ)+A(N,JJJ)-G*B(N,JJJ))/C
  DO17 I=2,N
C ***** SEGMENT 297 *****
  F=0.0
  F=0.0
  MEN=1-I
  DO18 J=2,I
C ***** SEGMENT 298 *****
  L=M+J-I
  F=E+Y(L)*A(M,L)-Z(L)*B(M,L)
C ***** SEGMENT 299 *****
  18 F=F+Y(L)*B(M,L)+Z(L)*A(M,L)
C ***** SEGMENT 300 *****
  E=A(I,M)+A(M,M)+B(M,M)*B(I,M)
  G=Y(I)
  Y(M)=(G-E)+A(M,M)+(Z(M)-F)*B(M,M)/D
C ***** SEGMENT 301 *****
  17 Z(M)=(Z(M)-F)*A(M,M)-(G-E)*B(M,M)/D
C ***** SEGMENT 302 *****
  F=0.0
  DO51 I=1,N
C ***** SEGMENT 303 *****
  F=Y(I)*Y(I)+Z(I)+Z(I)
C ***** SEGMENTS 304 THRU 305 *****
  IF (F.GE.F) GO TO 51
C ***** SEGMENT 306 *****
  52 F=F
  L=I
C ***** SEGMENT 307 *****
  51 CONTINUE

```

130


```

      F=Y(I)*Y(I)+Z(I)*Z(I)
C***** SEGMENTS 331 THRU 332 *****
      IF (E.GL.F) GO TO 22
C***** SEGMENT 333 *****
      23 F=F
      L=I
C***** SEGMENT 334 *****
      22 CONTINUE
C***** SEGMENT 335 *****
      A1=Y(L)
      A2=Z(L)
      DO 24 I=1,N
C***** SEGMENT 336 *****
      G=Y(I)
      Y(I)=(G*A1+Z(I)*A2)/E
C***** SEGMENT 337 *****
      24 Z(I)=(Z(I)*A1-G*A2)/F
C      NORMALIZED EIGENVECTOR OBTAINED
C PUT Y(I) (REAL PART) IN EIGVECT(NUMOFFV,I)
C***** SEGMENT 338 *****
      DO 100 I=1,N
C***** SEGMENT 339 *****
      100 EIGVECT(NUM,I)=Y(I)
C***** SEGMENT 340 *****
      NUM=NUM+1
C PUT Z(I) (COMPLEX PART) IN EIGVECT(NUMOFFV,I)
      DO 101 I=1,N
C***** SEGMENT 341 *****
      101 EIGVECT(NUM,I)=Z(I)
C***** SEGMENT 342 *****
      RETURN
      END

```

END PASG1: VALUE= 127

ORIGINAL PAGE IS
OF POOR QUALITY

LOCAL ANALYSIS--

COUNT		CODE
177 OCCURRENCES OF	<ASSIGNMENT> X=...	36
112 OCCURRENCES OF	<ASSIGNMENT> X(...)=...	36
8 OCCURRENCES OF	CALL X(...)	38
23 OCCURRENCES OF	CONTINUE	55
1 OCCURRENCES OF	DATA /.../	11
6 OCCURRENCES OF	DIMENSION X(...),...	18
1 OCCURRENCES OF	DIMENSION X(...)	18
65 OCCURRENCES OF	DO	37
6 OCCURRENCES OF	END	56
1 OCCURRENCES OF	FORMAT	41
76 OCCURRENCES OF	GOTO X	59
55 OCCURRENCES OF	IF(<IF>)<STATEMENT> ALSO IF(<SWITCH>)...	45
	<STATEMENT> DATA FOR BOOLEAN IF ARE--	
	54 OCCURRENCES OF CODE 59 <STATEMENT>S.	
	1 OCCURRENCES OF CODE 61 <STATEMENT>S.	
	0 ILL FORMED <STATEMENT>S AND	
	0 IMPROPER <STATEMENT>S.	
7 OCCURRENCES OF	RETURN	61
5 OCCURRENCES OF	SUBROUTINE X(...)	01
2 OCCURRENCES OF	WRITE (...)	42
48 OCCURRENCES OF	COMMENTS	
595 OCCURRENCES OF	STATEMENTS	
158 OCCURRENCES OF	LABELLED STATEMENTS	

16c

SEGMENT EXECUTION FREQUENCIES

X	0	1	2	3	4	5	6	7	8	9
0	-----	1	2	3	3	2	1	3	2	2
10	1	1	3	3	1	4	4	2	3	0
20	3	8	3	2	3	0	3	12	3	2
30	1	4	16	1	2	3	1	4	4	1
40	3	9	9	3	3	0	3	2	1	3
50	1	1	1	0	1	1	4	1	1	0
60	3	2	1	2	9	20	20	2	18	9
70	7	2	7	7	0	7	7	7	0	0
80	0	0	0	0	7	0	7	0	0	0
90	0	7	7	3	4	3	7	5	2	5
100	7	1	0	1	4	16	64	4	0	4
110	4	1	3	0	0	0	0	1	4	3
120	1	2	1	1	0	1	1	1	0	1
130	1	1	0	1	1	0	1	7	2	5
140	18	7	18	18	18	0	18	18	33	33
150	18	7	18	33	33	18	18	7	2	5
160	18	5	0	0	0	0	0	0	0	0
170	0	2	0	0	0	0	0	0	0	0
180	0	0	0	0	0	0	0	0	0	0
190	0	0	0	0	0	0	0	0	0	0
200	0	0	0	0	0	4	16	4	12	6
210	6	18	18	6	6	12	0	12	24	12
220	4	16	16	0	16	4	16	4	16	48
230	96	96	48	48	20	28	48	16	64	16
240	4	12	36	18	18	36	12	4	8	12
250	8	8	4	4	8	4	16	8	8	16
260	4	16	4	16	4	0	0	0	0	0
270	0	0	0	0	0	0	0	0	0	0
280	0	0	0	0	0	0	0	0	0	0
290	0	0	0	0	0	0	0	0	0	0
300	0	0	0	0	0	0	0	0	0	0
310	0	0	0	0	0	0	0	0	0	0
320	0	0	0	0	0	0	0	0	0	0
330	0	0	0	0	0	0	0	0	0	0
340	0	0	0	0	0	0	0	0	0	0

OF 11

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
1N	PROGRAM UNSEIG(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)		
2 C	FOR DESCRIPTION OF METHOD USED IN THIS PROGRAM, SEE **THE OR		
3 C	TRANSFORMATION** BY FRANCIS, COMPUTER JOURNAL, VOL. 4 PP. 265,332		
4 C	START HERE TO PRODUCE UPPER HESSENBURG MATRIX		
5	DIMENSION A(10,10),VEC(8),EIGVEC(4,4),INT(10),CC(10),SS(10)		
6	DIMENSION Z(10),D(10,10),BB(10,10)		
	DATA A / 0.1,5.8,1.9,3.2,6*0.0,2.8,3.9,0.7,0.1,6*0.0,		
8	X 6.2,3.5,0.6,1.2,6*0.0,9.9,6.2,1.7,3.8 /		
9	NMAX = 10	1	
10	NN = 4	1	
11	NVEC = 4	1	
12	N=NN	1 MIN= 4 MAX= 4	
		FIRST= 4 LAST= 4	
13	K=N-2	1 MIN= 2 MAX= 2	
		FIRST= 2 LAST= 2	
14	DO1J=1,K	1 MIN1= 2 MAX1= 2	
15	K1=J+1	2 MIN= 2 MAX= 3	
		FIRST= 2 LAST= 3	
16	K2=J+2	2 MIN= 3 MAX= 4	
		FIRST= 3 LAST= 4	
17	GREAT= ABS(A(K1,J))	2 MIN= .9392687E+00 MAX= .5800000E+01	
		FIRST= .58000E+01 LAST= .93927E+00	
18	MAX=K1	2 MIN= 2 MAX= 3	
		FIRST= 2 LAST= 3	
19	DO2I=K2,N	2 MIN1= 3 MAX1= 4	
		MIN2= 4 MAX2= 4	
20	AB= ABS(A(I,J))	3 MIN= .1900000E+01 MAX= .3200000E+01	
		FIRST= .19000E+01 LAST= .20819E+01	
21	IF (GREAT.GE.AB) GO TO 2	3 TRUE 2 FALSE 1	
22	3 GREAT=AB	1 MIN= .2081926E+01 MAX= .2081926E+01	
		FIRST= .20819E+01 LAST= .20819E+01	
23	MAX=I	1 MIN= 4 MAX= 4	
		FIRST= 4 LAST= 4	
24	2 CONTINUE	3	
25 C	INT(I) GIVES INFORMATION ABOUT INTERCHANGES		
26	INT(I)=MAX	2 MIN= 2 MAX= 4	
		FIRST= 2 LAST= 4	
27	IF (MAX.EQ.K1) GO TO 5	2 TRUE 1 FALSE 1	
28	4 DOAL=J,N	1 MIN1= 2 MAX1= 2	
		MIN2= 4 MAX2= 4	
29	E=A(K1,L)	3 MIN= -.9392687E+00 MAX= -.3310345E+00	
		FIRST= -.93927E+00 LAST= -.33103E+00	
30	A(K1,L)=A(MAX,L)	3 MIN= -.2081926E+01 MAX= .3793103E+00	
		FIRST= -.20819E+01 LAST= .37931E+00	
31	6 A(MAX,L)=E	3 MIN= -.9392687E+00 MAX= -.3310345E+00	
		FIRST= -.93927E+00 LAST= -.33103E+00	
32	DO7L=1,N	1 MIN1= 4 MAX1= 4	
33	E=A(L,K1)	4 MIN= -.7310345E+00 MAX= .6200000E+01	
		FIRST= .62000E+01 LAST= -.54655E+00	
34	A(L,K1)=A(L,MAX)	4 MIN= -.3310345E+00 MAX= .9900000E+01	
		FIRST= .99000E+01 LAST= -.33103E+00	
35	7 A(L,MAX)=E	4 MIN= -.7310345E+00 MAX= .6200000E+01	
		FIRST= .62000E+01 LAST= -.54655E+00	
36 C	INTERCHANGE DONE		
37	5 DO8I=K2,N	2 MIN1= 3 MAX1= 4	
		MIN2= 4 MAX2= 4	

ORIGINAL PAGE IS
OF POOR QUALITY

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNING)	COUNT	SPECIFIC EXECUTION DATA
38	IF (A(I,J).EQ.0.) GO TO 8	3 TRUE 0 FALSE 3	
39	A(I,J)=-A(I,J)/A(K1,J)	3 MIN= -.5517241E+00 MAX= -.3275862E+00 FIRST= -.32759E+00 LAST= -.45115E+00	
40	DO 11 M=K1,N	3 MIN1= 2 MAX1= 3 MIN2= 4 MAX2= 4	
41	11 A(I,M)=A(I,M)+A(I,J)*A(K1,M)	8 MIN= -.2051724E+01 MAX= .3793103E+00 FIRST= -.57759E+00 LAST= -.21674E+00	
42	8 CONTINUE	3	
43 C	LEFT MULT. DONE		
44	DO 2000 M=K2,N	2 MIN1= 3 MAX1= 4 MIN2= 4 MAX2= 4	
45	IF (A(M,J).EQ.0.) GO TO 2000	3 TRUE 0 FALSE 3	
46	10 DO 2001 I=1,M	3 MIN1= 4 MAX1= 4	
47	2001 A(I,K1)=A(I,K1)-A(I,M)*A(M,J)	12 MIN= -.2291201E+01 MAX= .1269715E+02 FIRST= .48310E+01 LAST= -.59999E+00	
48	2000 CONTINUE	3	
49	1 CONTINUE	2	
50 C	RIGHT MULT. DONE		
51 C	HESSBURG MATRIX NOW FORMED AS A(I,J). MULTIPLIERS STORED WITH IT		
52	DO 105 I=1,N	1 MIN1= 4 MAX1= 4 4 MIN1= 4 MAX1= 4	
53	DO 105 J=1,N	16 MIN= -.2081926E+01 MAX= .1269715E+02 FIRST= .10000E+00 LAST= -.21674E+00	
54	105 D(I,J)=A(I,J)		
55 C	PUT APPROPRIATE ZEROS WHERE MULTIPLIERS WERE STORED		
56	DO 106 I=3,N	1 MIN1= 4 MAX1= 4	
57	L=I-2	2 MIN= 1 MAX= 2 FIRST= 1 LAST= 2	
58	DO 106 J=1,L	2 MIN1= 1 MAX1= 2	
59	106 A(I,J)=0.0	3	
60	K1=M+1	1 MIN= 5 MAX= 5 FIRST= 5 LAST= 5	
61	R=0.0	1	
62	C=0.0	1	
63	DO 33 I=1,M	1 MIN1= 4 MAX1= 4	
64	R=R+ ABS(A(I,I))	4 MIN= .1000000E+00 MAX= .2929026E+02 FIRST= .10000E+00 LAST= .29290E+02	
65	33 C=C+ ABS(A(I,N))	4 MIN= .6200000E+01 MAX= .1064778E+02 FIRST= .62000E+01 LAST= .10648E+02	
66	DO 34 I=2,N	1 MIN1= 4 MAX1= 4	
67	RS=0.0	3	
68	CS=0.0	3	
69	L=I-1	3 MIN= 1 MAX= 3 FIRST= 1 LAST= 3	
70	DO 35 J=L,N	3 MIN1= 1 MAX1= 3 MIN2= 4 MAX2= 4	
71	RS=RS+ ABS(A(I,J))	4 MIN= .5999461E+00 MAX= .2554628E+02 FIRST= .58000E+01 LAST= .81669E+00	
72	K1J=K1-I	9 MIN= 1 MAX= 4 FIRST= 4 LAST= 1	
73	K1I=K1-I	9 MIN= 1 MAX= 3 FIRST= 3 LAST= 1	
74	35 CS=CS+ ABS(A(K1J,K1I))	9 MIN= .5999461E+00 MAX= .2112564E+02 FIRST= .59995E+00 LAST= .59000E+01	
75	IF (RS.LE.R) GO TO 36	3 TRUE 2 FALSE 1	
76	37 R=RS	0	
77	36 IF (CS.LE.C) GO TO 34	3 TRUE 2 FALSE 1	
78	44 C=CS	1 MIN= .2112564E+02 MAX= .2112564E+02	

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA		
79	34 CONTINUE	3			
80	IJK=1	1			
81	IF (R.GE.C) GO TO 54	1 TRUE	1 FALSE		0
82	C DELTA IS CRITERION FOR STOPPING AN ITERATION				
83	53 DELTA=R*1.0E-11	* 0			
84	C DEL IS ARBITRARY SHIFT				
85	DEL=R/17.1829	* 0			
86	GO TO 153	* 0			
87	54 DELTA=C*1.0E-11	1 MIN= .2112564E-09 MAX= .2112564E-09 FIRST= .21126E-09 LAST= .21126E-09			
88	DEL=C/17.1829	1 MIN= .1229457E+01 MAX= .1229457E+01 FIRST= .12295E+01 LAST= .12295E+01			
89	C DELTA COMPUTED				
90	C PRODUCE A-PI, ITERATE ON THIS TO BREAK UP EIGENVALUES OF EQUAL MODULUS				
91	153 DO 107 I=1,N	1 MIN1= 4 MAX1= 4			
92	107 A(I,I)=A(I,I)-DEL	4 MIN= -.1446200E+01 MAX= .7237784E+01 FIRST= -.11295E+01 LAST= -.14462E+01			
93	55 IF (N.NE.1) GO TO 48	1 TRUE 1 FALSE 0			
94	C SUBROUTINE CALCEV CALCULATES EIGENVALUES OF 1X1 OR 2X2 MATRICES.				
95	C PI IS ADDED BACK IN CALCEV				
96	47 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)	* 0			
97	GO TO 76	* 0			
98	48 IF (N.NE.2) GO TO 80	3 TRUE 2 FALSE 1			
99	49 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)	1			
100	GO TO 76	1			
101	C QR TRANSFORMATION STARTS HERE				
102	C INITIALIZE FOR DETERMINING SUBSEQUENT SHIFTS				
103	80 AR1=0.0	2			
104	AR3=0.0	2			
105	AR4=0.0	2			
106	C RE-ENTER AFTER REALIT OR COMPT SUBROUTINES CALLED				
107	C SELECT FIRST ((ZERO)) SUB-DIAGONAL ELEMENT FROM BOTTOM				
108	61 L=N	9 MIN= 3 MAX= 4 FIRST= 4 LAST= 3			
109	D056 I=2,N	9 MIN1= 3 MAX1= 4			
110	N2I=N+2-1	20 MIN= 2 MAX= 4 FIRST= 4 LAST= 3			
111	IF(ABS(A(N2I,N2I-1)).LT.DELTA)GO TO 57	20 TRUE 2 FALSE 18			
112	56 L=L-1	18 MIN= 1 MAX= 3 FIRST= 3 LAST= 1			
113	C VALUE OF L SELECTED				
114	57 IF (L.NE.N) GO TO 59	9 TRUE 7 FALSE 2			
115	58 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)	2			
116	N=N-1	2 MIN= 2 MAX= 3 FIRST= 3 LAST= 2			
117	GO TO 48	2			
118	59 IF ((L-N+1).NE.0) GO TO 75	7 TRUE 7 FALSE 0			
119	60 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)	* 0			
120	N=N-2	* 0			
121	GO TO 55	* 0			

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
122 C	DETERMINE EIGENVALUES OF 2X2 MATRIX -- ROW 1 = A(N-1,N-1) A(N-1,N)		
123 C	ROW 2 = A(N,N-1) A(N,N)		
124 C	AND BASE SHIFT ON THESE		
125	75 B=A(N-1,N-1)+A(N,N)	7	MIN= -.5384725E+01 MAX= -.3040462E-02 FIRST= -.26262E+01 LAST= -.49193E+01
126	C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)	7	MIN= -.8740759E+01 MAX= .1267871E+01 FIRST= .12679E+01 LAST= -.81379E+01
127	C1=A(N-1,N-1)-A(N,N)	7	MIN= -.8004470E+01 MAX= .2654877E+01 FIRST= .26624E+00 LAST= -.75333E+01
128	DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)	7	MIN= .1825211E+01 MAX= .6395829E+02 FIRST= .18252E+01 LAST= .56751E+02
129	IF (DISC.GE.0.) GO TO 63	7	TRUE 7 FALSE 0
130	62 R1=0.5*B	*	0
131	R2=.5* SQRT(ABS(DISC))	*	0
132	ANUM=(R1-AR3)*(R1-AR3)+(R2-AR4)*(R2-AR4)	*	0
133	DENOM=AR3*AR3+AR4*AR4	*	0
134	IF (DENOM.NE.0.) GO TO 200	*	0
135	201 RHO=0.0	*	0
136	SIGMA=0.0	*	0
137	GO TO 203	*	0
138	200 RHO=C	*	0
139	SIGMA=B	*	0
140	203 AR3=R1	*	0
141	AR4=R2	*	0
142	CALL COMPIT(A,NMAX,L,N,RHO,SIGMA)	*	0
143	GO TO 61	*	0
144	63 IF (B.GE.0.) GO TO 65	7	TRUE 0 FALSE 7
145	64 R1=.5*(B- SQRT(DISC))	7	MIN= -.6691059E+01 MAX= -.1328918E+01 FIRST= -.19886E+01 LAST= -.62263E+01
146	GO TO 66	7	
147	65 R1=.5*(B+ SQRT(DISC))	*	0
148	IF (R1.NE.0.) GO TO 66	*	0
149	83 R2=0.0	*	0
150	GO TO 84	*	0
151	66 R2=C/R1	7	MIN= -.6375759E+00 MAX= .1325918E+01 FIRST= -.63758E+00 LAST= .13070E+01
152 C	ROOTS OF 2*2 MATRIX FOUND		
153	84 IF(ABS(A(N,N)-R1).GE. ABS(A(N,N)-R2))GO TO 68	7	TRUE 3 FALSE 4
154	67 AR2=R1	4	MIN= -.1988580E+01 MAX= -.1328918E+01 FIRST= -.19886E+01 LAST= -.13290E+01
155	GO TO 69	4	
156	68 AR2=R2	3	MIN= .1301875E+01 MAX= .1307015E+01 FIRST= .13019E+01 LAST= .13070E+01
157	69 IF (AR1.NE.0.) GO TO 71	7	TRUE 5 FALSE 2
158	72 SHIFT=0.0	2	
159	GO TO 73	2	
160	71 SHIFT=AR2	5	MIN= -.1337816E+01 MAX= .1307015E+01 FIRST= -.13378E+01 LAST= .13070E+01
161	73 AR1=AR2	7	MIN= -.1988580E+01 MAX= .1307015E+01 FIRST= -.19886E+01 LAST= .13070E+01
162	CALL REALIT(A,NMAX,L,N,SHIFT,CC,SS)	7	
163	GO TO 61	7	
164 C	ARE VECTORS NEEDED		

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
165	76 IF (NVEC.EQ.0) GO TO 600	1 TRUE	0 FALSE 1
166	C INVERSE ITERATION IS USED FOR EIGENVECTORS		
167	550 SMALL=DELTA*1.0E-30	1 MIN= .2112564E-39 MAX= .2112564E-39 FIRST= .21126E-39 LAST= .21126E-39	
168	NUM=0	1	
169	JI=1	1	
170	NM=NM+NM	1 MIN= 8 MAX= 8 FIRST= 8 LAST= 8	
171	C C1 IS REAL PART OF EIGENVALUE		
172	513 C1=VEC(JI)	4 MIN= -.5247020E+01 MAX= .1121005E+02 FIRST= -.99502E-01 LAST= .11210E+02	
173	C C2 IS COMPLEX PART OF EIGENVALUE		
174	C2=VEC(JI+1)	4 MIN= 0. MAX= 0. FIRST= 0. LAST= 0.	
175	NUM=NUM+1	4 MIN= 1 MAX= 4 FIRST= 1 LAST= 4	
176	C REPLACE HESSENBURG MATRIX AND MULTIPLIERS		
177	DO 570 I=1,NM	4 MIN1= 4 MAX1= 4	
178	DO 570 J=1,NM	16 MIN1= 4 MAX1= 4	
179	570 A(I,J)=D(I,J)	64 MIN= -.2081926E+01 MAX= .1269715E+02 FIRST= .10000E+00 LAST= -.21674E+00	
180	IF (C2.NE.0.) GO TO 561	4 TRUE 0 FALSE 4	
181	CALL RVEC(A,NMAX,INT,C1,SMALL,NM,NUM,EIGVEC,NVEC,CC,SS)	4	
182	JI=JI+2	4 MIN= 3 MAX= 9 FIRST= 3 LAST= 9	
183	IF(NVEC.EQ.NUM)GO TO 600	4 TRUE 1 FALSE 3	
184	GO TO 513	3	
185	561 CALL IMVEC(A,NMAX,INT,C1,C2,SMALL,NM,NUM,EIGVEC,NVEC,CC,SS,Z,BB)	0	
186	C NEXT STATEMENT IGNORES CONJUGATE COMPLEX EIGENVALUES		
187	JI=JI+4	0	
188	IF(NVEC.EQ.NUM)GO TO 600	0	
189	GO TO 513	0	
190	600 CONTINUE	1	
191	END	1	

ORIGINAL PAGE IS
OF POOR QUALITY

* PROGRAM EVALUATOR AND TESTER REPORT *
*
* FOR SUBROUTINE UNSEIG *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 190

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	156	82.1
NONEXECUTABLE SOURCE	4	2.1
COMMENT	30	15.8
NONSTANDARD	1	.5
BRANCH	58	N/A
CALL	8	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT 560

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	125	80.1
BRANCH	35	60.3
CALL	4	50.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

60

STATEMENT NUMBER	PROGRAM LISTING (LEADING W INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
1	SUBROUTINE CALCEV(A,NMAX,DEL,VEC,IJK,N,M)		
2	DIMENSION A(NMAX,HMAX),VEC(1)		
3	IF (M.NE.1) GO TO 2	3 TRUE 1 FALSE 2	
4	1 A1=0.0	2	
5	A(N,N)=A(N,N)+DEL	2 MIN= -.9950155E+01 MAX= .2536472E+01 FIRST= -.99502E+01 LAST= .25365E+01	
6	C STORE EIGENVALUES IN VEC		
7	VEC(IJK)=A(N,N)	2 MIN= -.9950155E+01 MAX= .2536472E+01 FIRST= -.99502E+01 LAST= .25365E+01	
8	VEC(IJK+1)=A1	2 MIN= 0. MAX= 0. FIRST= 0. LAST= 0.	
9	IJK=IJK+2	2 MIN= 3 MAX= 5 FIRST= 3 LAST= 5	
10	RETURN	2	
11	2 B=A(N-1,N-1)+A(N,N)	1 MIN= .3504115E+01 MAX= .3504115E+01 FIRST= .35041E+01 LAST= .35041E+01	
12	C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)	1 MIN= -.6463907E+02 MAX= -.6463907E+02 FIRST= -.64639E+02 LAST= -.64639E+02	
13	C1=A(N-1,N-1)-A(N,N)	1 MIN= .1681400E+02 MAX= .1681400E+02 FIRST= .16814E+02 LAST= .16814E+02	
14	DISC=C1+C1+.0*A(N,N-1)*A(N-1,N)	1 MIN= .2708351E+03 MAX= .2708351E+03 FIRST= .27084E+03 LAST= .27084E+03	
15	IF (DISC.LT.0.) GO TO 9	1 TRUE 0 FALSE 1	
16	4 A1=0.0	1	
17	IF (B.GE.0.) GO TO 6	1 TRUE 1 FALSE 0	
18	R1=0.5*(B- SQRT(DISC))	0	
19	GO TO 7	0	
20	6 R1=0.5*(B+ SQRT(DISC))	1 MIN= .9980592E+01 MAX= .9980592E+01 FIRST= .99806E+01 LAST= .99806E+01	
21	IF (R1.NE.0.) GO TO 7	1 TRUE 1 FALSE 0	
22	10 R2=0.0	0	
23	GO TO 11	0	
24	7 R2=C/R1	1 MIN= -.6476477E+01 MAX= -.6476477E+01 FIRST= -.64765E+01 LAST= -.64765E+01	
25	11 R1=R1+DEL	1 MIN= .1121005E+02 MAX= .1121005E+02 FIRST= .11210E+02 LAST= .11210E+02	
26	R2=R2+DEL	1 MIN= -.5247020E+01 MAX= -.5247020E+01 FIRST= -.52470E+01 LAST= -.52470E+01	
27	VEC(IJK)=R2	1 MIN= -.5247020E+01 MAX= -.5247020E+01 FIRST= -.52470E+01 LAST= -.52470E+01	
28	VEC(IJK+1)=A1	1 MIN= 0. MAX= 0. FIRST= 0. LAST= 0.	
29	VEC(IJK+2)=R1	1 MIN= .1121005E+02 MAX= .1121005E+02 FIRST= .11210E+02 LAST= .11210E+02	
30	VEC(IJK+3)=A1	1 MIN= 0. MAX= 0. FIRST= 0. LAST= 0.	
31	GO TO 8	1	
32	9 R1=0.5*B	0	
33	A1=-.5* SQRT(ABS(DISC))	0	
34	R1=R1+DEL	0	
35	VEC(IJK)=R1	0	
36	VEC(IJK+1)=-A1	0	
37	VEC(IJK+2)=R1	0	

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE CALCEV *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS

41

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	37	90.2
NONEXECUTABLE SOURCE	3	7.3
COMMENT	1	2.4
NONSTANDARD	0	0.0
BRANCH	11	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT

34

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	26	70.3
BRANCH	6	54.5
CALL	0	100.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

ORIGINAL PAGE IS
OF POOR QUALITY

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
1	SUBROUTINE REALIT(A,NMAX,L,N,SHIFT,C,S)		
2	DIMENSION A(NMAX,NMAX),C(1),S(1)		
3	IF (SHIFT.EQ.0.) GO TO 2		
4	1 DO3 I=L,N	7 TRUE 5 MIN1=	2 FALSE 1 MAX1=
5	3 A(I,I)=A(I,I)-SHIFT	MIN2=	3 MAX2=
6	2 M=N-1	18 MIN=	1173958E+02
7	DO4 I=L,M	FIRST=	52505E+01 LAST=
8	DENOM= SQRT(A(I,I)**2+A(I+1,I)**2)	7 MIN=	2 MAX=
9	IF (DENOM.GT.0.) GO TO 10	FIRST=	3 LAST=
10	11 C(I)=1.0	7 MIN1=	1 MAX1=
11	S(I)=0.0	MIN2=	2 MAX2=
12	GO TO 4	18 MIN=	1456132E+01 MAX=
13	10 C(I)=A(I,I)/DENOM	FIRST=	59089E+01 LAST=
14	S(I)=A(I+1,I)/DENOM	18 TRUE	18 FALSE
15	A(I,I)=DENOM	0	0
16	12 II=I+1	0	0
17	DO 35 J=II,N	18 MIN=	1000000E+01 MAX=
18	F=A(I,J)	FIRST=	19114E+00 LAST=
19	A(I,J)=A(I+1,J)*S(I)+F*C(I)	18 MIN=	4120136E+00 MAX=
20	A(I+1,J)=A(I+1,J)*C(I)-F*S(I)	FIRST=	9815621E+00
21	35 CONTINUE	18 MIN=	98156E+00 LAST=
22	4 CONTINUE	FIRST=	59089E+01 LAST=
23	C TRIANGULAR MATRIX FORMED	18 MIN=	2 MAX=
24	DO5 J=L,M	FIRST=	2 LAST=
25	DO6 I=L,J	18 MIN1=	2 MAX1=
26	F=A(I,J)	MIN2=	3 MAX2=
27	A(I,J)=A(I,J+1)*S(J)+F*C(J)	33 MIN=	1394996E+02 MAX=
28	6 A(I,J+1)=A(I,J+1)*C(J)-F*S(J)	FIRST=	10293E+02 LAST=
29	A(J+1,J)=A(J+1,J+1)*S(J)	33 MIN=	1345768E+02 MAX=
30	5 A(J+1,J+1)=A(J+1,J+1)*C(J)	FIRST=	51369E+01 LAST=
31	C TRANSFORMATION COMPLETED	33 MIN=	1394996E+02 MAX=
32	IF (SHIFT.EQ.0.) RETURN	FIRST=	11487E+02 LAST=
33	7 DO9 I=L,N	33	18
34	9 A(I,I)=A(I,I)+SHIFT	7 MIN1=	1 MAX1=
35	8 RETURN	MIN2=	2 MAX2=
		18 MIN1=	1 MAX1=
		MIN2=	1 MAX2=
		33 MIN=	1411128E+02 MAX=
		FIRST=	59089E+01 LAST=
		33 MIN=	7961958E+01 MAX=
		FIRST=	39127E+01 LAST=
		33 MIN=	1411128E+02 MAX=
		FIRST=	67819E+01 LAST=
		18 MIN=	4605441E+00 MAX=
		FIRST=	11459E+02 LAST=
		18 MIN=	2653969E+01 MAX=
		FIRST=	22314E+01 LAST=
		7 TRUE	2 FALSE
		5 MIN1=	1 MAX1=
		MIN2=	3 MAX2=
		18 MIN=	6925412E+01 MAX=
		FIRST=	69542E+01 LAST=

100

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE REALIT *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 34

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	31	86.1
NONEXECUTABLE SOURCE	3	8.3
COMMENT	2	5.6
NONSTANDARD	0	0.0
BRANCH	7	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT 515

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	28	90.3
BRANCH	5	71.4
CALL	0	100.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
1	SUBROUTINE COMPIT(A,NMAX,L,N,RHO,SIGMA)		
2	DIMENSION A(NMAX,NMAX)		
3	C HOUSEHOLDER TYPE MATRICES USED INSTEAD OF ROTATIONS(SEE FRANCIS PAPER)		
4	M=N-1	*	0
5	DO 1 I=L,M	*	0
6	IF (I.NE.L) GO TO 3	*	0
7	2 B1=A(L,L)*(A(L,L)-SIGMA)+A(L,L+1)*A(L+1,L)+RHO	*	0
8	B2=A(L+1,L)*(A(L,L)+A(L+1,L+1)-SIGMA)	*	0
9	B3=A(L+1,L)*A(L+2,L+1)	*	0
10	A(L+2,L)=0.0	*	0
11	GO TO 6	*	0
12	3 B1=A(I,I-1)	*	0
13	B2=A(I+1,I-1)	*	0
14	IF (I.EQ.M) GO TO 5	*	0
15	4 B3=A(I+2,I-1)	*	0
16	GO TO 6	*	0
17	5 B3=0.0	*	0
18	6 D= SQRT(B1*B1+B2*B2+B3*B3)	*	0
19	IF (B1.GE.0.) GO TO 8	*	0
20	7 D=-D	*	0
21	8 IF (D.EQ.0.) GO TO 1	*	0
22	9 C1=B2/(D+B1)	*	0
23	C2=B3/(D+B1)	*	0
24	E=2.0/(1.0+C1+C1+C2*C2)	*	0
25	DO 10 K=1,N	*	0
26	IF (I.EQ.M) GO TO 12	*	0
27	11 G=E*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))	*	0
28	A(I,K)=A(I,K)-G	*	0
29	A(I+1,K)=A(I+1,K)-C1*G	*	0
30	A(I+2,K)=A(I+2,K)-C2*G	*	0
31	GO TO 10	*	0
32	12 G=F*(A(I,K)+C1*A(I+1,K))	*	0
33	A(I,K)=A(I,K)-G	*	0
34	A(I+1,K)=A(I+1,K)-C1*G	*	0
35	10 CONTINUE	*	0
36	IF (I.EQ.L) GO TO 14	*	0
37	13 A(I,I-1)=-D	*	0
38	A(I+1,I-1)=0.0	*	0
39	IF (I.EQ.M) GO TO 14	*	0
40	15 A(I+2,I-1)=0.0	*	0
41	C ROW OPERATION COMPLETED		
42	14 J=I+2	*	0
43	IF (J.LE.N) GO TO 17	*	0
44	16 J=N	*	0
45	17 DO 18 K=L,J	*	0
46	IF (I.EQ.M) GO TO 20	*	0
47	19 G=E*(A(K,I)+C1*A(K,I+1)+C2*A(K,I+2))	*	0
48	A(K,I)=A(K,I)-G	*	0
49	A(K,I+1)=A(K,I+1)-C1*G	*	0
50	A(K,I+2)=A(K,I+2)-C2*G	*	0
51	GO TO 18	*	0
52	20 G=E*(A(K,I)+C1*A(K,I+1))	*	0
53	A(K,I)=A(K,I)-G	*	0
54	A(K,I+1)=A(K,I+1)-C1*G	*	0
55	18 CONTINUE	*	0
56	J=I+3	*	0

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
57	IF (J.GT.N) GO TO 1	*	0
58	21 G=E+C2+A(I+3,I+2)	*	0
59	A(I+3,I)=-G	*	0
60	A(I+3,I+1)=-C1*G	*	0
61	A(I+3,I+2)=A(I+3,I+1)-C2*G	*	0
62 C	COLUMN OPERATION COMPLETED		
63	1 CONTINUE	*	0
64	RETURN	*	0
65	END		

ORIGINAL PAGE IS
OF POOR QUALITY

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE COMPIT *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 65

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	59	90.8
NONEXECUTABLE SOURCE	3	4.6
COMMENT	3	4.6
NONSTANDARD	0	0.0
BRANCH	24	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

THIS SUBROUTINE WAS NOT EXECUTED

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNING)	COUNT	SPECIFIC EXECUTION DATA
1	SUBROUTINE RVEC(A,NMAX,INT,C1,SMALL,N,NUM,EIGVEC,NVEC,INTER,Y)		
2	DIMENSION A(NMAX,NMAX),INTER(1),Y(1),INT(1),EIGVEC(NMAX,NMAX)		
3	DO1 I=1,N	4	MIN1= 4 MAX1= 4
4	1 A(I,I)=A(I,I)-C1	16	MIN= -.1142679E+02 MAX= .1371426E+02 FIRST= .19950E+00 LAST= -.11427E+02
5	DO2 I=2,N	4	MIN1= 4 MAX1= 4
6	IF(ABS(A(I-1,I-1)).GE. ABS(A(I,I-1)))GO TO 4	12	TRUE 6 FALSE 6
7	3 INTER(I)=1	6	MIN= 2 MAX= 4 FIRST= 2 LAST= 4
8	J=I-1	6	MIN= 1 MAX= 3 FIRST= 1 LAST= 3
9	DO5 K=J,N	6	MIN1= 1 MAX1= 3 MIN2= 4 MAX2= 4
10	B=A(I-1,K)	18	MIN= -.3365121E+01 MAX= .1269715E+02 FIRST= .19950E+00 LAST= .46004E+01
11	A(I-1,K)=A(I,K)	18	MIN= -.1142679E+02 MAX= .1371426E+02 FIRST= .58000E+01 LAST= -.11427E+02
12	5 A(I,K)=B	18	MIN= -.3365121E+01 MAX= .1269715E+02 FIRST= .19950E+00 LAST= .46004E+01
13	GO TO 6	6	
14	4 INTER(I)=I-1	6	MIN= 1 MAX= 3 FIRST= 2 LAST= 2
15	6 IF (A(I,I-1).EQ.0.) GO TO 2	12	TRUE 0 FALSE 12
16	C=-A(I,I-1)/A(I-1,I-1)	12	MIN= -.9218999E+00 MAX= .7913944E+00 FIRST= -.34397E-01 LAST= .40259E+00
17	A(I,I-1)=C	12	MIN= -.9218999E+00 MAX= .7913944E+00 FIRST= -.34397E-01 LAST= .40259E+00
18	DO 8 K=I,N	12	MIN1= 2 MAX1= 4 MIN2= 4 MAX2= 4
19	8 A(I,K)=A(I,K)+C*A(I-1,K)	24	MIN= -.3365121E+01 MAX= .1596498E+02 FIRST= .99984E+01 LAST= .10311E-09
20	2 CONTINUE	12	
21	C TRIANGULAR MATRIX FORMED		
22	DO9 I=1,N	4	MIN1= 4 MAX1= 4
23	IF(ABS(A(I,I)).GE.SMALL)GO TO 9	16	TRUE 16 FALSE 0
24	10 A(I,I)=SMALL	0	
25	9 CONTINUE	16	
26	C NORMALISED ((EIGENVECTOR)) OF HESSENBURG MATRIX OBTAINED FOR RHS = 1		
27	DO 5555 I=1,N	4	MIN1= 4 MAX1= 4
28	5555 Y(I)=1.0	16	
29	DO 8888 JJJ=1,4	4	
30	7777 Y(N)=Y(N)/A(N,N)	16	MIN= -.3574285E+14 MAX= .1330028E+13 FIRST= -.35743E+14 LAST= .87749E+10
31	B=Y(N)	16	MIN= -.3574285E+14 MAX= .1330028E+13 FIRST= -.35743E+14 LAST= .87749E+10
32	DO11 I=2,N	16	MIN1= 4 MAX1= 4
33	E=0.0	48	
34	M=N+1-I	48	MIN= 1 MAX= 3 FIRST= 3 LAST= 1
35	DO12 J=2,I	48	MIN1= 2 MAX1= 4
36	L=N+J-I	96	MIN= 2 MAX= 4 FIRST= 4 LAST= 4
37	12 E=E+Y(L)*A(M,L)	96	MIN= -.1304838E+15 MAX= .1661351E+15 FIRST= -.19119E+14 LAST= .71225E+13
38	Y(M)=(Y(M)-E)/A(M,M)	48	MIN= -.8752666E+13 MAX= .305043E+14 FIRST= .69849E+13 LAST= .64109E+12

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
39	IF(ABS(B).GE. ABS(Y(M)))GO TO 11	48 TRUE 20 FALSE 28	
40	13 B=Y(M)	28 MIN= -.0752668E+13 MAX= .7828595E+13 FIRST= .78286E+13 LAST= .89285E+12	
41	11 CONTINUE	48	
42	D014 I=1,N	16 MIN1= 4 MAX1= 4	
43	14 Y(I)=Y(I)/B	64 MIN= -.8944238E+00 MAX= .1000000E+01 FIRST= .13774E+00 LAST= .98280E-02	
44	IF(IJJJ.EQ.4)GO TO 6666	16 TRUE 4 FALSE 12	
45	D015 I=2,N	12 MIN1= 4 MAX1= 4	
46	IF (INTER(I).NE.1) GO TO 15	36 TRUE 18 FALSE 18	
47	16 B=Y(I)	18 MIN= -.3651198E+00 MAX= .1000000E+01 FIRST= -.36512E+00 LAST= .98280E-02	
48	Y(I)=Y(I-1)	18 MIN= -.1301137E+01 MAX= .1000000E+01 FIRST= .19794E+00 LAST= .90085E+00	
49	Y(I-1)=B	18 MIN= -.3651198E+00 MAX= .1000000E+01 FIRST= -.36512E+00 LAST= .98280E-02	
50	15 Y(I)=Y(I)+A(I,I-1)*Y(I-1)	36 MIN= -.1312023E+01 MAX= .1374845E+01 FIRST= .21050E+00 LAST= .90481E+00	
51	8088 CONTINUE	12	
52	C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED (NOT NORMALIZED)		
53	6666 M=N-1	4 MIN= 3 MAX= 3 FIRST= 3 LAST= 3	
54	D021 I=2,M	4 MIN1= 3 MAX1= 3	
55	J=N+1-I	8 MIN= 2 MAX= 3 FIRST= 3 LAST= 2	
56	L=J+1	8 MIN= 3 MAX= 4 FIRST= 4 LAST= 3	
57	D020 K=L,N	8 MIN1= 3 MAX1= 4 MIN2= 4 MAX2= 4	
58	20 Y(K)=Y(K)-Y(J)*A(K,J-1)	12 MIN= -.3968651E+00 MAX= .9118357E+00 FIRST= .91184E+00 LAST= .36454E+00	
59	LL=INT(M-I)	8 MIN= 2 MAX= 4 FIRST= 4 LAST= 2	
60	IF (J.EQ.LL) GO TO 21	8 TRUE 4 FALSE 4	
61	19 B=Y(J)	4 MIN= -.8944238E+00 MAX= -.1364325E+00 FIRST= -.19542E+00 LAST= -.18719E+00	
62	Y(J)=Y(LL)	4 MIN= -.2086210E+00 MAX= .9118357E+00 FIRST= .91184E+00 LAST= -.74622E-01	
63	Y(LL)=B	4 MIN= -.8944238E+00 MAX= -.1364325E+00 FIRST= -.19542E+00 LAST= -.18719E+00	
64	21 CONTINUE	8	
65	C UNNORMALIZED EIGENVECTOR OF ORIGINAL MATRIX OBTAINED		
66	B=0.0	4	
67	D022 I=1,M	4 MIN1= 4 MAX1= 4	
68	IF(ABS(B).GE. ABS(Y(I)))GO TO 22	16 TRUE 8 FALSE 8	
69	23 B=Y(I)	8 MIN= -.3651198E+00 MAX= .1000000E+01 FIRST= .19794E+00 LAST= .10000E+01	
70	22 CONTINUE	16	
71	D024 I=1,M	4 MIN1= 4 MAX1= 4	
72	24 Y(I)=Y(I)/B	16 MIN= -.5009484E+00 MAX= .1000000E+01 FIRST= .24986E+00 LAST= .36454E+00	
73	C NORMALIZED EIGENVECTOR OBTAINED		

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
---------------------	---	-------	-------------------------

74	C PUT Y(I) (REAL EIGENVECTOR) IN EIGVEC(NUMDEFY,I)		
----	--	--	--

75	DO 100 I=1,N		
----	--------------	--	--

76	100 EIGVEC(NUM,I)=Y(I)		
----	------------------------	--	--

N	MINI=	N	MAXI=	N
16	MIN= -.5009484E+00	MAX=	.1000000E+01	
	FIRST= .24986E+00	LAST=	.36454E+00	

77	RETURN		
----	--------	--	--

78	END		
----	-----	--	--

ORIGINAL PAGE IS
OF POOR QUALITY

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE AVEC *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS

78

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	69	88.5
NONEXECUTABLE SOURCE	3	3.8
COMMENT	6	7.7
NONSTANDARD	0	0.0
BRANCH	17	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT

1238

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	68	98.6
BRANCH	15	88.2
CALL	0	100.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

18D

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
	SUBROUTINE IMVEC(A,NMAX,INT,C1,AMU,SMALL,N,NUM,EIGVEC,NVEC,INTER,		
2	1Y,Z,B)		
	DIMENSION A(NMAX,NMAX),INTER(1),Y(1),Z(1),INT(1),B(NMAX,NMAX),		
4	1EIGVEC(NMAX,NMAX)		
5	DO50 I=1,N	*	0
6	DO50 J=1,N	*	0
7	50 B(I,J)=0.0	*	0
8	DO1 I=1,N	*	0
9	A(I,1)=A(I,1)-C1	*	0
10	1 B(I,1)=B(I,1)-AMU	*	0
11	DO2 I=2,N	*	0
12	C=A(I-1,I-1)*A(I-1,I-1)+B(I-1,I-1)*B(I-1,I-1)	*	0
13	D=A(I,I-1)*A(I,I-1)+B(I,I-1)*B(I,I-1)	*	0
14	IF (C.GE.D) GO TO 4	*	0
15	3 INTER(I)=I	*	0
16	G=C	*	0
17	C=D	*	0
18	D=B	*	0
19	J=I-1	*	0
20	DO5 K=J,N	*	0
21	G=A(I-1,K)	*	0
22	A(I-1,K)=A(I,K)	*	0
23	A(I,K)=G	*	0
24	G=B(I-1,K)	*	0
25	B(I-1,K)=B(I,K)	*	0
26	5 B(I,K)=G	*	0
27	GO TO 6	*	0
28	4 INTER(I)=I-1	*	0
29	6 IF (D.EQ.0.) GO TO 2	*	0
30	7 G=A(I,I-1)	*	0
31	A(I,I-1)=-(G*A(I-1,I-1)+B(I,I-1)*B(I-1,I-1))/C	*	0
32	B(I,I-1)=-(B(I,I-1)*A(I-1,I-1)-G*B(I-1,I-1))/C	*	0
33	DO8 K=I,N	*	0
34	A(I,K)=A(I,K)+A(I,I-1)*A(I-1,K)-B(I,I-1)*B(I-1,K)	*	0
35	8 B(I,K)=B(I,K)+B(I,I-1)*A(I-1,K)+A(I,I-1)*B(I-1,K)	*	0
36	2 CONTINUE	*	0
37 C	TRIANGULAR MATRIX FORMED		
38	DO9 I=1,N	*	0
39	C=A(I,1)*A(I,1)+B(I,1)*B(I,1)	*	0
40	IF (C.GE.SMALL) GO TO 9	*	0
41	A(I,1)=SQRT(SMALL)	*	0
42	B(I,1)=0.0	*	0
43	9 CONTINUE	*	0
44	DO 1111 I=1,N	*	0
45	Y(I)=1.0	*	0
46	1111 Z(I)=0.0	*	0
47	DO 8088 JJJ=1,N	*	0
48	C=A(N,N)*A(N,N)+B(N,N)*B(N,N)	*	0
49	G=Y(N)	*	0
50	Y(N)=(G*A(N,N)+Z(N)*B(N,N))/C	*	0
51	Z(N)=(Z(N)*A(N,N)-G*B(N,N))/C	*	0
52	DO17 J=2,N	*	0
53	E=0.0	*	0
54	F=0.0	*	0
55	M=N+1-I	*	0
56	DO18 J=2,I	*	0

STATEMENT NUMBER	PROGRAM LISTING (LEADING H INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
57	L=N+J-1	*	0
58	E=E+Y(L)*A(M,L)-Z(L)*B(M,L)	*	0
59	18 F=F+Y(L)*B(M,L)+Z(L)*A(M,L)	*	0
60	D=A(M,M)*A(M,M)+B(M,M)*B(M,M)	*	0
61	G=Y(M)	*	0
62	Y(M)=((G-E)*A(M,M)+(Z(M)-F)*B(M,M))/D	*	0
63	17 Z(M)=((Z(M)-F)*A(M,M)-(G-E)*B(M,M))/D	*	0
64	E=0.0	*	0
65	D051 I=1,N	*	0
66	F=Y(I)*Y(I)+Z(I)*Z(I)	*	0
67	IF (E.GE.F) GO TO 51	*	0
68	52 E=F	*	0
69	L=I	*	0
70	51 CONTINUE	*	0
71	A1=Y(L)	*	0
72	A2=Z(L)	*	0
73	D014 I=1,N	*	0
74	G=Y(I)	*	0
75	Y(I)=(G*A1+Z(I)*A2)/E	*	0
76	14 Z(I)=(Z(I)+A1-G*A2)/E	*	0
77	IF (J.J.EQ.4) GO TO 6666	*	0
78	D015 I=2,N	*	0
79	IF (INTER(I).NE.I) GO TO 30	*	0
80	16 G=Y(I)	*	0
81	Y(I)=Y(I-1)	*	0
82	Y(I-1)=G	*	0
83	G=Z(I)	*	0
84	Z(I)=Z(I-1)	*	0
85	Z(I-1)=G	*	0
86	30 Y(I)=Y(I)+A(I,I-1)*Y(I-1)-B(I,I-1)*Z(I-1)	*	0
87	15 Z(I)=Z(I)+B(I,I-1)*Y(I-1)+A(I,I-1)*Z(I-1)	*	0
88	8888 CONTINUE	*	0
89	C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED	*	0
90	6666 M=N-1	*	0
91	D021 I=2,M	*	0
92	J=N+1-I	*	0
93	L=J+1	*	0
94	D020 K=L,N	*	0
95	Y(K)=Y(K)-Y(J)*A(K,J-1)+Z(J)*B(K,J-1)	*	0
96	20 Z(K)=Z(K)-Z(J)*A(K,J-1)+Y(J)*B(K,J-1)	*	0
97	LL=INT(N-I)	*	0
98	IF (J.EQ.LL) GO TO 21	*	0
99	19 G=Y(J)	*	0
100	Y(J)=Y(LL)	*	0
101	Y(LL)=G	*	0
102	G=Z(J)	*	0
103	Z(J)=Z(LL)	*	0
104	Z(LL)=G	*	0
105	21 CONTINUE	*	0
106	C UNNORMALIZED EIGENVECTOR OBTAINED	*	0
107	E=0.0	*	0
108	D022 I=1,N	*	0
109	F=Y(I)*Y(I)+Z(I)*Z(I)	*	0
110	IF (E.GE.F) GO TO 22	*	0
111	23 E=F	*	0
112	L=I	*	0

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
113	22 CONTINUE	*	0
114	A1=Y(L)	*	0
115	A2=Z(L)	*	0
116	DO 24 I=1,N	*	0
117	G=Y(I)	*	0
118	Y(I)=(G*A1+Z(I)*A2)/E	*	0
119	24 Z(I)=(Z(I)*A1-G*A2)/E	*	0
120 C	NORMALIZED EIGENVECTOR OBTAINED		
121 C	PUT Y(I) (REAL PART) IN EIGVECT(NUMOFEV,I)		
122	DO 100 I=1,N	*	0
123	100 EIGVEC(NUM,I)=Y(I)	*	0
124	NUM=NUM+1	*	0
125 C	PUT Z(I) (COMPLEX PART) IN EIGVECT(NUMOFEV,I)		
126	DO 101 I=1,N	*	0
127	101 EIGVEC(NUM,I)=Z(I)	*	0
128	RETURN	*	0
129	END		

ORIGINAL PAGE IS
OF POOR QUALITY

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE IMVEC *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 127

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	118	92.9
NONEXECUTABLE SOURCE	3	2.4
COMMENT	6	4.7
NONSTANDARD	0	0.0
BRANCH	17	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

THIS SUBROUTINE WAS NOT EXECUTED

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR PROGRAM UNSEIG *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 537

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	470	87.5
NONEXECUTABLE SOURCE	19	3.5
COMMENT	48	8.9
NONSTANDARD	1	.2
BRANCH	13%	N/A
CALL	8	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT 2347

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	247	52.6
BRANCH	61	45.5
CALL	4	50.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

* PROGRAM EVALUATOR AND TESTER REPORT *

* SUBROUTINE OPERATIONAL SUMMARY *

SUBROUTINE NAME	NO OF STATEMENTS EXECUTED	PERCENT EXECUTED	NO OF CALLS EXECUTED	PERCENT EXECUTED	NO OF BRANCHES EXECUTED	PERCENT EXECUTED
UNSEIG	125	80.1	4	50.0	35	60.3
CALCEV	26	70.3	0	100.0	6	54.5
REALTT	20	90.3	0	100.0	5	71.4
RVEC	68	98.6	0	100.0	15	88.2
PROGRAM	247	52.6	4	50.0	61	45.5

SUBROUTINES NOT EXECUTED

COMFIT
INVEC

* PROGRAM EVALUATOR AND TESTER REPORT *
* SUBROUTINE EXECUTION SUMMARY *

SUBROUTINE NAME	NUMBER TIMES ENTERED	TOTAL EXECUTION COUNT	TOTAL CALL COUNT	PERCENT OF TOTAL EXECUTIONS	TOTAL BRANCH COUNT	PERCENT OF TOTAL EXECUTIONS	TOTAL I/O COUNT	PERCENT OF TOTAL EXECUTIONS
UNSEIG	1	560	14	2.50	121	21.61	0	0.00
CALCEV	3	34	0	0.00	7	29.59	0	0.00
REALIT	7	515	0	0.00	32	6.21	0	0.00
RVEC	4	1238	0	0.00	170	13.73	0	0.00

ORIGINAL PAGE IS
OF POOR QUALITY

* PROGRAM EVALUATOR AND TESTER REPORT *
* SUBROUTINE TIMING SUMMARY *

TOTAL EXECUTION TIME FOR THIS CASE .042 SEC
TOTAL TIME IN MONITORED ROUTINES .038 SEC
TOTAL TIME IN OTHER ROUTINES .004 SEC

DETAILED SUBROUTINE TIME MONITORING

SUBROUTINE NAME	TIME RELATIVE TO MAXIMUM SUBROUTINE EXECUTION TIME (* = TWO PERCENT)	ACTUAL TIME IN SECONDS	PERCENT OF TOTAL MONITORED TIME
UNSEIG	*****	.017	44.74
CALCEY	***	.001	2.63
REALIT	*****	.008	21.05
RVEC	*****	.012	31.58

TOTAL TIME IN POSTPROCESSOR 1.667 SEC

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA		
---------------------	---	-------	-------------------------	--	--

1N	PROGRAM UNSEIG(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)				
2 C	FOR DESCRIPTION OF METHOD USED IN THIS PROGRAM, SEE **THE QR				
3 C	TRANSFORMATION** BY FRANCIS, COMPUTER JOURNAL, VOL. 4 PP. 265,332				
4 C	START HERE TO PRODUCE UPPER HESSENBURG MATRIX				
5	DIMENSION A(10,10),VEC(8),EIGVEC(4,4),INT(10),CC(10),SS(10)				
6	DIMENSION Z(10),D(10,10),BB(10,10)				
	DATA A / 0.1,5.3,1.9,3.2,6*0.0,2.8,3.9,0.7,0.1,6*0.0,				
8	X 6.2,3.5,0.6,1.2,6*0.0,9.9,6.2,1.7,3.8 /				
9	NMAX = 10	1			
10	NN = 4	1			
11	NVEC = 4	1			
12	N=NN	1			
13	K=N-2	1			
14	DO1J=1,K	1			
15	K1=J+1	2			
16	K2=J+2	2			
17	GREAT=ABS(A(K1,J))	2			
18	MAX=K1	2			
19	DO2I=K2,N	2			
20	AB=ABS(A(I,J))	3			
21	IF (GREAT.GE.AB) GO TO 2	3 TRUE	2 FALSE	1	
22	3 GREAT=AB	1			
23	MAX=I	1			
24	2 CONTINUE	3			
25 C	INT(I) GIVES INFORMATION ABOUT INTERCHANGES				
26	INT(J)=MAX	2			
27	IF (MAX.EQ.K1) GO TO 5	2 TRUE	1 FALSE	1	
28	4 DO6L=J,N	1			
29	E=A(K1,L)	3			
30	A(K1,L)=A(MAX,L)	3			
31	6 A(MAX,L)=E	3			
32	DO7L=1,N	1			
33	E=A(L,K1)	4			
34	A(L,K1)=A(L,MAX)	4			
35	7 A(L,MAX)=E	4			
36 C	INTERCHANGE DONE				
37	5 DO8I=K2,N	2			
38	IF (A(I,J).EQ.0.) GO TO 8	3 TRUE	0 FALSE	3	
39	A(I,J)=-A(I,J)/A(K1,J)	3			
40	DO11M=K1,N	3			
41	11 A(I,M)=A(I,M)+A(I,J)*A(K1,M)	8			
42	8 CONTINUE	3			
43 C	LEFT MULT. DONE				
44	DO 2000 M=K2,N	2			
45	IF (A(M,J).EQ.0.) GO TO 2000	3 TRUE	0 FALSE	3	
46	10 DO 2001 I=1,N	3			
47	2001 A(I,K1)=A(I,K1)-A(I,M)*A(M,J)	12			
48	2000 CONTINUE	3			
49	1 CONTINUE	2			
50 C	RIGHT MULT. DONE				
51 C	HESSENBURG MATRIX NOW FORMED AS A(I,J). MULTIPLIERS STORED WITH IT				
52	DO 105 I=1,N	1			
53	DO 105 J=1,N	4			

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA		
54	105 DO 1, J)=A(I, J)	16			
55	C PUT APPROPRIATE ZEROS WHERE MULTIPLIERS WERE STORED				
56	DO 106 I=3, N	1			
57	L=I-2	2			
58	DO 106 J=1, L	2			
59	106 A(I, J)=0.0	3			
60	KI=I+1	1			
61	R=0.0	1			
62	C=0.0	1			
63	DO 33 I=1, N	1			
64	R=R+ ABS(A(I, I))	4			
65	33 C=C+ ABS(A(I, N))	4			
66	DO 34 I=2, N	1			
67	RS=0.0	3			
68	CS=0.0	3			
69	L=I-1	3			
70	DO 35 J=L, N	3			
71	RS=RS+ ABS(A(I, J))	9			
72	KIJ=KI-J	9			
73	KII=KI-1	9			
74	35 CS=CS+ ABS(A(KIJ, KII))	9			
75	IF (RS.LE.R) GO TO 36	3 TRUE	3 FALSE	0	
76	37 R=RS	0			
77	36 IF (CS.LE.C) GO TO 34	3 TRUE	2 FALSE	1	
78	44 C=CS	1			
79	34 CONTINUE	3			
80	IJK=1	1			
81	IF (R.GE.C) GO TO 54	1 TRUE	1 FALSE	0	
82	C DELTA IS CRITERION FOR STOPPING AN ITERATION				
83	53 DELTA=R+1.0E-11	0			
84	C DEL IS ARBITRARY SHIFT				
85	DEL=R/17.1829	0			
86	GO TO 153	0			
87	54 DELTA=C+1.0E-11	1			
88	DEL=C/17.1829	1			
89	C DELTA COMPLETED				
90	C PRODUCE A-PI, ITERATE ON THIS TO BREAK UP EIGENVALUES OF EQUAL MODULUS				
91	153 DO 107 I=1, N	1			
92	107 A(I, I)=A(I, I)-DEL	4			
93	55 IF (N.NE.1) GO TO 48	1 TRUE	1 FALSE	0	
94	C SUBROUTINE CALCEV CALCULATES EIGENVALUES OF 1X1 OR 2X2 MATRICES.				
95	C PI IS ADDED BACK IN CALCEV				
96	47 CALL CALCEV(A, NMAX, DEL, VEC, IJK, N, 1)	0			
97	GO TO 74	0			
98	48 IF (N.NE.2) GO TO 80	3 TRUE	2 FALSE	1	
99	49 CALL CALCEV(A, NMAX, DEL, VEC, IJK, N, 2)	1			
100	GO TO 74	1			
101	C OR TRANSFORMATION STARTS HERE				
102	C INITIALIZE FOR DETERMINING SUBSEQUENT SHIFTS				
103	80 AR1=0.0	2			
104	AR3=0.0	2			

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA		
105	AR4=0.0	2			
106	C RE-ENTER AFTER REALIT OR COMPIT SUBROUTINES CALLED				
107	C SELECT FIRST ((ZERO)) SUB-DIAGONAL ELEMENT FROM BOTTOM				
108	61 L=N	9			
109	D056 I=2,N	9			
110	N21=N+2-1	20			
111	IF(ABS(A(N21,N21-1)) .LT. DELTA) GO TO 57	20 TRUE	2 FALSE	16	
112	56 L=L-1	18			
113	C VALUE OF L SELECTED				
114	57 IF (L.NE.N) GO TO 59	9 TRUE	7 FALSE	2	
115	58 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)	2			
116	N=N-1	2			
117	GO TO 48	2			
118	59 IF ((L-N+1).NE.0) GO TO 75	7 TRUE	7 FALSE	0	
119	60 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)	*			
120	N=N-2	*			
121	GO TO 55	*			
122	C DETERMINE EIGENVALUES OF 2X2 MATRIX -- ROW 1 = A(N-1,N-1) A(N-1,N)				
123	C ROW 2 = A(N,N-1) A(N,N)				
124	C AND BASE SHIFT ON THESE				
125	75 B=A(N-1,N-1)+A(N,N)	7			
126	C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)	7			
127	C1=A(N-1,N-1)-A(N,N)	7			
128	DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)	7			
129	IF (DISC.GE.0.) GO TO 63	7 TRUE	7 FALSE	0	
130	62 R1=0.5*B	*			
131	R2=.5* SQRT(ABS(DISC))	*			
132	ANUM=(R1-AR3)*(R1-AR3)+(R2-AR4)*(R2-AR4)	*			
133	DENOM=AR3*AR3+AR4*AR4	*			
134	IF (DENOM.NE.0.) GO TO 200	*			
135	201 RHO=0.0	*			
136	SIGMA=0.0	*			
137	GO TO 203	*			
138	200 RHO=C	*			
139	SIGMA=B	*			
140	203 AR3=R1	*			
141	AR4=R2	*			
142	CALL COMPIT(A,NMAX,L,N,RHO,SIGMA)	*			
143	GO TO 61	*			
144	63 IF (B.GE.0.) GO TO 65	7 TRUE	0 FALSE	7	
145	64 R1=.5*(B- SQRT(DISC))	7			
146	GO TO 66	7			
147	65 R1=.5*(B+ SQRT(DISC))	*			
148	IF (R1.NE.0.) GO TO 66	*			
149	83 R2=0.0	*			
150	GO TO 84	*			
151	66 R2=C/R1	7			
152	C ROOTS OF 2*2 MATRIX FOUND				
153	84 IF(ABS(A(N,N)-R1).GE. ABS(A(N,N)-R2)) GO TO 68	7 TRUE	3 FALSE	4	
154	67 AR2=R1	4			
155	GO TO 69	4			

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA		
156	68 AR2=R2	3			
157	69 IF (AR1.NE.0.) GO TO 71	7 TRUE	5 FALSE	2	
158	72 SHIFT=0.0	2			
159	GO TO 73	2			
160	71 SHIFT=AR2	5			
161	73 AR1=AR2	7			
162	CALL REALIT(A,NMAX,L,N,SHIFT,CC,SS)	7			
163	GO TO 61	7			
164	C ARE VECTORS NEEDED				
165	76 IF (NVEC.EQ.0) GO TO 600	1 TRUE	0 FALSE	1	
166	C INVERSE ITERATION IS USED FOR EIGENVECTORS				
167	550 SMALL=DELTA+1.0E-30	1			
168	NUM=0	1			
169	JI=1	1			
170	NM=NN+NM	1			
171	C C1 IS REAL PART OF EIGENVALUE				
172	513 C1=VEC(JI)	4			
173	C C2 IS COMPLEX PART OF EIGENVALUE				
174	C2=VEC(JI+1)	4			
175	NUM=NUM+1	4			
176	C REPLACE HESSENBURG MATRIX AND MULTIPLIERS				
177	DO 570 I=1,NN	4			
178	DO 570 J=1,NN	16			
179	570 A(I,J)=D(I,J)	64			
180	IF (C2.NE.0.) GO TO 561	4 TRUE	0 FALSE	4	
181	CALL RVEC(A,NMAX,INT,C1,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS)	4			
182	JI=JI+2	4			
183	IF(NVEC.EQ.NUM)GO TO 600	4 TRUE	1 FALSE	3	
184	GO TO 513	3			
185	561 CALL INVEC(A,NMAX,INT,C1,C2,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS,Z,BB)	* 0			
186	C NEXT STATEMENT IGNORES CONJUGATE COMPLEX EIGENVALUES				
187	JI=JI+4	* 0			
188	IF(NVEC.EQ.NUM)GO TO 600	* 0			
189	GO TO 513	* 0			
190	600 CONTINUE	1			
191	END	1			

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE UNSEIG *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 190

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	156	82.1
NONEXECUTABLE SOURCE	4	2.1
COMMENT	30	15.8
NONSTANDARD	1	.5
BRANCH	58	N/A
CALL	8	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

ORIGINAL PAGE IS
OF POOR QUALITY

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT 560

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	125	80.1
BRANCH	35	60.3
CALL	4	50.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA		
1	SUBROUTINE CALCEV(A,NMAX,DEL,VEC,IJK,N,M)				
2	DIMENSION A(NMAX,NMAX),VEC(1)				
3	IF (M.NE.1) GO TO 2	3 TRUE	1 FALSE	2	
4	1 AI=0.0	2			
5	A(N,N)=A(N,N)+DEL	2			
6	C STORE EIGENVALUES IN VEC				
7	VEC(IJK)=A(N,N)	2			
8	VEC(IJK+1)=AI	2			
9	IJK=IJK+2	2			
10	RETURN	2			
11	2 B=A(N-1,N-1)+A(N,N)	1			
12	C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)	1			
13	C1=A(N-1,N-1)-A(N,N)	1			
14	DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)	1			
15	IF (DISC.LT.0.) GO TO 9	1 TRUE	0 FALSE	1	
16	4 AI=0.0	1			
17	IF (B.GE.0.) GO TO 6	1 TRUE	1 FALSE	0	
18	R1=0.5*(B- SQRT(DISC))	0			
19	GO TO 7	0			
20	6 R1=0.5*(B+ SQRT(DISC))	1			
21	IF(R1.NE.0.) GO TO 7	1 TRUE	1 FALSE	0	
22	10 R2=0.0	0			
23	GO TO 11	0			
24	7 R2=C/R1	1			
25	11 R1=R1+DEL	1			
26	R2=R2+DEL	1			
27	VEC(IJK)=R2	1			
28	VEC(IJK+1)=AI	1			
29	VEC(IJK+2)=R1	1			
30	VEC(IJK+3)=AI	1			
31	GO TO 8	1			
32	9 R1=0.5*B	0			
33	AI=-.5* SQRT(ABS(DISC))	0			
34	R1=R1+DEL	0			
35	VEC(IJK)=R1	0			
36	VEC(IJK+1)=-AI	0			
37	VEC(IJK+2)=R1	0			
38	VEC(IJK+3)=AI	0			
39	8 IJK=IJK+4	1			
40	RETURN	1			
41	END				

6E

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE CALCEV *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS

41

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	37	90.2
NONEXECUTABLE SOURCE	3	7.3
COMMENT	1	2.4
NONSTANDARD	0	0.0
BRANCH	11	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT

34

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	26	70.3
BRANCH	6	54.5
CALL	0	100.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA		
1	SUBROUTINE REAL(TLA,NMAX,L,N,SHIFT,C,S)				
2	DIMENSION A(NMAX,NMAX),C(1),S(1)				
3	IF (SHIFT.EQ.0.) GO TO 2	7 TRUE	2 FALSE	5	
4	1 DO3 I=L,N	5			
5	3 A(I,I)=A(I,I)-SHIFT	18			
6	2 M=N-1	7			
7	DO4 I=L,M	7			
8	DENOM= SORT(A(I,I)**2+A(I+1,I)**2)	18			
9	IF (DENOM.GT.0.) GO TO 10	18 TRUE	18 FALSE	0	
10	11 C(I)=1.0	0			
11	S(I)=0.0	0			
12	GO TO 4	0			
13	10 C(I)=A(I,I)/DENOM	18			
14	S(I)=A(I+1,I)/DENOM	18			
15	A(I,I)=DENOM	18			
16	12 II=I+1	18			
17	DO 35 J=II,N	18			
18	F=A(I,J)	33			
19	A(I,J)=A(I+1,J)*S(I)+F*C(I)	33			
20	A(I+1,J)=A(I+1,J)*C(I)-F*S(I)	33			
21	35 CONTINUE	33			
22	4 CONTINUE	18			
23	C TRIANGULAR MATRIX FORMED				
24	DO5 J=L,M	7			
25	DO6 I=L,J	18			
26	F=A(I,J)	33			
27	A(I,J)=A(I,J+1)*S(J)+F*C(J)	33			
28	6 A(I,J+1)=A(I,J+1)*C(J)-F*S(J)	33			
29	A(J+1,J)=A(J+1,J+1)*S(J)	18			
30	5 A(J+1,J+1)=A(J+1,J+1)*C(J)	18			
31	C TRANSFORMATION COMPLETED				
32	IF (SHIFT.EQ.0.) RETURN	7 TRUE	2 FALSE	5	
33	7 DO9 I=L,N	5			
34	9 A(I,I)=A(I,I)+SHIFT	18			
35	8 RETURN	5			
36	END				

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE REALIT *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 36

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	31	86.1
NONEXECUTABLE SOURCE	3	8.3
COMMENT	2	5.6
NONSTANDARD	0	0.0
BRANCH	7	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT 515

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	28	90.3
BRANCH	5	71.4
CALL	0	100.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

ORIGINAL PAGE IS
OF POOR QUALITY

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
1	SUBROUTINE COMPIT(A,NMAX,L,N,RHO,SIGMA)		
2	DIMENSION A(NMAX,NMAX)		
3	C HOUSEHOLDER TYPE MATRICES USED INSTEAD OF ROTATIONS(SEE FRANCIS PAPER)		
4	M=N-1	*	0
5	DO 1 I=L,M	*	0
6	IF (I.NE.L) GO TO 3	*	0
7	2 B1=A(L,L)*(A(L,L)-SIGMA)+A(L,L+1)*A(L+1,L)+RHO	*	0
8	B2=A(L+1,L)*(A(L,L)+A(L+1,L+1)-SIGMA)	*	0
9	B3=A(L+1,L)*A(L+2,L+1)	*	0
10	A(L+2,L)=0.0	*	0
11	GO TO 6	*	0
12	3 B1=A(I,I-1)	*	0
13	B2=A(I+1,I-1)	*	0
14	IF (I.EQ.M) GO TO 5	*	0
15	4 B3=A(I+2,I-1)	*	0
16	GO TO 6	*	0
17	5 B3=0.0	*	0
18	6 D= SQRT(B1*B1+B2*B2+B3*B3)	*	0
19	IF (B1.GE.0.) GO TO 8	*	0
20	7 D=-D	*	0
21	8 IF (D.EQ.0.) GO TO 1	*	0
22	9 C1=B2/(D+B1)	*	0
23	C2=B3/(D+B1)	*	0
24	E=2.0/((1.0+C1+C1+C2+C2)	*	0
25	DO 10 K=I,N	*	0
26	IF (I.EQ.M) GO TO 12	*	0
27	11 G=E*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))	*	0
28	A(I,K)=A(I,K)-G	*	0
29	A(I+1,K)=A(I+1,K)-C1*G	*	0
30	A(I+2,K)=A(I+2,K)-C2*G	*	0
31	GO TO 10	*	0
32	12 G=E*(A(I,K)+C1*A(I+1,K))	*	0
33	A(I,K)=A(I,K)-G	*	0
34	A(I+1,K)=A(I+1,K)-C1*G	*	0
35	10 CONTINUE	*	0
36	IF (I.EQ.L) GO TO 14	*	0
37	13 A(I,I-1)=-D	*	0
38	A(I+1,I-1)=0.0	*	0
39	IF (I.EQ.M) GO TO 14	*	0
40	15 A(I+2,I-1)=0.0	*	0
41	C ROW OPERATION COMPLETED		
42	14 J=I+2	*	0
43	IF (J.LE.N) GO TO 17	*	0
44	16 J=N	*	0
45	17 DO 18 K=L,J	*	0
46	IF (I.EQ.M) GO TO 20	*	0
47	19 G=E*(A(K,I)+C1*A(K,I+1)+C2*A(K,I+2))	*	0
48	A(K,I)=A(K,I)-G	*	0
49	A(K,I+1)=A(K,I+1)-C1*G	*	0
50	A(K,I+2)=A(K,I+2)-C2*G	*	0
51	GO TO 18	*	0
52	20 G=E*(A(K,I)+C1*A(K,I+1))	*	0
53	A(K,I)=A(K,I)-G	*	0
54	A(K,I+1)=A(K,I+1)-C1*G	*	0
55	18 CONTINUE	*	0
56	J=I+3	*	0

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
57	IF (J.GT.N) GO TO 1	*	0
58	21 G=E+C2*A(I+3,I+2)	*	0
59	A(I+3,I)=G	*	0
60	A(I+3,I+1)=C1+G	*	0
61	A(I+3,I+2)=A(I+3,I+2)-C2*G	*	0
62 C	COLUMN OPERATION COMPLETED		
63	1 CONTINUE	*	0
64	RETURN	*	0
65	END		

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE COMPIT *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 65

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	59	90.8
NONEXECUTABLE SOURCE	3	4.6
COMMENT	3	4.6
NONSTANDARD	0	0.0
BRANCH	24	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

THIS SUBROUTINE WAS NOT EXECUTED

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA		
1	SUBROUTINE RVEC(A,NMAX,INT,C1,SMALL,N,NUM,EIGVEC,NVEC,INTER,Y)				
2	DIMENSION A(NMAX,NMAX),INTER(1),Y(1),INT(1),EIGVEC(NMAX,NMAX)				
3	DO1 I=1,N	4			
4	1 A(I,1)=A(I,1)-C1	16			
5	DO2 I=2,N	4			
6	IF(ABS(A(I-1,I-1)).GE.ABS(A(I,I-1)))GO TO 4	12	TRUE	6	FALSE 6
7	3 INTER(I)=I	6			
8	J=I-1	6			
9	DO3 K=J,N	6			
10	B=A(I-1,K)	18			
11	A(I-1,K)=A(I,K)	18			
12	5 A(I,K)=B	18			
13	GO TO 6	6			
14	4 INTER(I)=I-1	6			
15	6 IF(A(I,I-1).EQ.0.)GO TO 2	12	TRUE	0	FALSE 12
16	C=-A(I,I-1)/A(I-1,I-1)	12			
17	A(I,I-1)=C	12			
18	DO 8 K=I,N	12			
19	8 A(I,K)=A(I,K)+C*A(I-1,K)	24			
20	2 CONTINUE	12			
21	C - TRIANGULAR MATRIX FORMED				
22	DO9 I=1,N	4			
23	IF(ABS(A(I,1)).GE.SMALL)GO TO 9	16	TRUE	16	FALSE 0
24	10 A(I,1)=SMALL	0			
25	9 CONTINUE	16			
26	C NORMALISED ((EIGENVECTOR)) OF HESSENBURG MATRIX OBTAINED FOR RHS = 1				
27	DO 5555 I=1,N	4			
28	5555 Y(I)=1.0	16			
29	DO 8888 JJJJ=1,4	4			
30	7777 Y(N)=Y(N)/A(N,N)	16			
31	B=Y(N)	16			
32	DO11 I=2,N	16			
33	E=0.0	48			
34	M=N+1-I	48			
35	DO12 J=2,I	48			
36	L=N+J-I	96			
37	12 E=E+Y(L)*A(M,L)	96			
38	Y(M)=(Y(M)-E)/A(M,M)	48			
39	IF(ABS(B).GE.ABS(Y(M)))GO TO 11	48	TRUE	20	FALSE 28
40	13 B=Y(M)	28			
41	11 CONTINUE	48			
42	DO14 I=1,N	16			
43	14 Y(I)=Y(I)/B	64			
44	IF(JJJJ.EQ.4)GO TO 6666	16	TRUE	4	FALSE 12
45	DO15 I=2,N	12			
46	IF(INTER(I).NE.I)GO TO 15	36	TRUE	18	FALSE 18
47	16 B=Y(I)	18			
48	Y(I)=Y(I-1)	18			
49	Y(I-1)=B	18			
50	15 Y(I)=Y(I)+A(I,I-1)*Y(I-1)	36			
51	8888 CONTINUE	12			

ORIGINAL PAGE IS
OF POOR QUALITY

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA		
52 C EIGENVECTOR OF HESSENBORG MATRIX OBTAINED (NOT NORMALIZED)					
53	6666 M=N-1	4			
54	0021 I=2,M	4			
55	J=N+1-I	2			
56	L=J+	8			
57	0020 K=L,N	8			
58	20 Y(K)=Y(K)-Y(J)*A(K,J-1)	12			
59	LL=INT(N-1)	8			
60	IF (J,EQ,LL) GO TO 21	8	TRUE	4 FALSE	4
61	19 B=Y(J)	4			
62	Y(J)=Y(LL)	4			
63	Y(LL)=B	4			
64	21 CONTINUE	8			
65 C UNNORMALIZED EIGENVECTOR OF ORIGINAL MATRIX OBTAINED					
66	B=0.0	4			
67	0022 I=1,N	4			
68	IF(ABS(B).GE. ABS(Y(I)))GO TO 22	16	TRUE	8 FALSE	8
69	23 B=Y(I)	8			
70	22 CONTINUE	16			
71	0024 I=1,N	4			
72	24 Y(I)=Y(I)/B	16			
73 C NORMALIZED EIGENVECTOR OBTAINED					
74 C PUT Y(I) (REAL EIGENVECTOR) IN EIGVECT(NUMOFEV,I)					
75	DO 100 I=1,N	4			
76	100 EIGVECT(NUM,I)=Y(I)	16			
77	RETURN	4			
78	END				

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE RVEC *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 78

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	69	88.5
NONEXECUTABLE SOURCE	3	3.8
COMMENT	6	7.7
NONSTANDARD	0	0.0
BRANCH	17	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT 1238

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	68	98.6
BRANCH	15	88.2
CALL	0	100.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
SUBROUTINE IMVEC(A,NMAX,INT,C1,AMU,SMALL,N,NUM,EIGVEC,NVEC,INTER,			
2	1Y,Z,B)		
	DIMENSION A(NMAX,NMAX),INTER(1),Y(1),Z(1),INT(1),B(NMAX,NMAX),		
4	1EIGVEC(NMAX,NMAX)		
5	DO50 I=1,N	*	0
6	DO50 J=1,N	*	0
7	50 B(I,J)=0.0	*	0
8	DO1 I=1,N	*	0
9	A(I,I)=A(I,I)-C1	*	0
10	1 B(I,I)=B(I,I)-AMU	*	0
11	DO2 I=2,N	*	0
12	C=A(I-1,I-1)*A(I-1,I-1)+B(I-1,I-1)*B(I-1,I-1)	*	0
13	D=A(I,I-1)*A(I,I-1)+B(I,I-1)*B(I,I-1)	*	0
14	IF (C.GE.D) GO TO 4	*	0
15	3 INTER(I)=I	*	0
16	G=C	*	0
17	C=D	*	0
18	D=G	*	0
19	J=I-1	*	0
20	DO5 K=J,N	*	0
21	G=A(I-1,K)	*	0
22	A(I-1,K)=A(I,K)	*	0
23	A(I,K)=G	*	0
24	G=B(I-1,K)	*	0
25	B(I-1,K)=B(I,K)	*	0
26	5 B(I,K)=G	*	0
27	GO TO 6	*	0
28	4 INTER(I)=I-1	*	0
29	6 IF (D.EQ.0.) GO TO 2	*	0
30	7 G=A(I,I-1)	*	0
31	A(I,I-1)=-(G*A(I-1,I-1)+B(I,I-1)*B(I-1,I-1))/C	*	0
32	B(I,I-1)=-(B(I,I-1)*A(I-1,I-1)+G*B(I-1,I-1))/C	*	0
33	DO8 K=I,N	*	0
34	A(I,K)=A(I,K)+A(I,I-1)*A(I-1,K)-B(I,I-1)*B(I-1,K)	*	0
35	8 B(I,K)=B(I,K)+B(I,I-1)*A(I-1,K)+A(I,I-1)*B(I-1,K)	*	0
36	2 CONTINUE	*	0
37	C TRIANGULAR MATRIX FORMED		
38	DO9 I=1,N	*	0
39	C=A(I,I)+A(I,I)*B(I,I)+B(I,I)	*	0
40	IF (C.GE.SMALL) GO TO 9	*	0
41	A(I,I)=SQRT(SMALL)	*	0
42	B(I,I)=0.0	*	0
43	9 CONTINUE	*	0
44	DO 1111 I=1,N	*	0
45	Y(I)=1.0	*	0
46	1111 Z(I)=0.0	*	0
47	DO 8888 JJJ=1,4	*	0
48	C=A(N,N)+A(N,N)+B(N,N)+B(N,N)	*	0
49	G=Y(N)	*	0
50	Y(N)=(G*A(N,N)+Z(N)*B(N,N))/C	*	0
51	Z(N)=(Z(N)*A(N,N)-G*B(N,N))/C	*	0
52	DO17 I=2,N	*	0
53	E=0.0	*	0
54	F=0.0	*	0
55	M=N+1-I	*	0
56	DO18 J=2,I	*	0

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
57	L=N+J-1	*	0
58	E=E+Y(L)*A(M,L)-Z(L)*B(M,L)	*	0
59	18 F=F+Y(L)*B(M,L)+Z(L)*A(M,L)	*	0
60	D=A(M,M)*A(M,M)+B(M,M)*B(M,M)	*	0
61	G=Y(M)	*	0
62	Y(M)=((Q-E)*A(M,M)+(Z(M)-F)*B(M,M))/D	*	0
63	17 Z(M)=((Z(M)-F)*A(M,M)-(G-E)*B(M,M))/D	*	0
64	E=0.0	*	0
65	D051 I=1,N	*	0
66	F=Y(I)*Y(I)+Z(I)*Z(I)	*	0
67	IF (E.GE.F) GO TO 51	*	0
68	52 E=F	*	0
69	L=I	*	0
70	51 CONTINUE	*	0
71	A1=Y(L)	*	0
72	A2=Z(L)	*	0
73	D014 I=1,N	*	0
74	G=Y(I)	*	0
75	Y(I)=(G+A1+Z(I)*A2)/E	*	0
76	14 Z(I)=(Z(I)+A1-G*A2)/E	*	0
77	IF (JJJ.EQ.4) GO TO 6666	*	0
78	D015 I=2,N	*	0
79	IF (INTER(I).NE.1) GO TO 30	*	0
80	16 G=Y(I)	*	0
81	Y(I)=Y(I-1)	*	0
82	Y(I-1)=G	*	0
83	G=Z(I)	*	0
84	Z(I)=Z(I-1)	*	0
85	Z(I-1)=G	*	0
86	30 Y(I)=Y(I)+A(I,I-1)*Y(I-1)-B(I,I-1)*Z(I-1)	*	0
87	15 Z(I)=Z(I)+B(I,I-1)*Y(I-1)+A(I,I-1)*Z(I-1)	*	0
88	8888 CONTINUE	*	0
89	C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED		
90	6666 M=N-1	*	0
91	D021 I=2,M	*	0
92	J=N+1-I	*	0
93	L=J+1	*	0
94	D020 K=L,N	*	0
95	Y(K)=Y(K)-Y(J)*A(K,J-1)+Z(J)*B(K,J-1)	*	0
96	20 Z(K)=Z(K)-Z(J)*A(K,J-1)+Y(J)*B(K,J-1)	*	0
97	LL=INT(N-1)	*	0
98	IF (J.EQ.LL) GO TO 21	*	0
99	19 G=Y(J)	*	0
100	Y(J)=Y(LL)	*	0
101	Y(LL)=G	*	0
102	G=Z(J)	*	0
103	Z(J)=Z(LL)	*	0
104	Z(LL)=G	*	0
105	21 CONTINUE	*	0
106	C UNNORMALIZED EIGENVECTOR OBTAINED		
107	E=0.0	*	0
108	D022 I=1,N	*	0
109	F=Y(I)*Y(I)+Z(I)*Z(I)	*	0
110	IF (E.GE.F) GO TO 22	*	0
111	23 E=F	*	0
112	L=I	*	0

ORIGINAL PAGE IS
OF POOR QUALITY

STATEMENT NUMBER	PROGRAM LISTING (LEADING N INDICATES CONVERSION WARNINGS)	COUNT	SPECIFIC EXECUTION DATA
113	22 CONTINUE	*	0
114	A1=Y(L)	*	0
115	A2=Z(L)	*	0
116	DO24 I=1,N	*	0
117	G=Y(I)	*	0
118	Y(I)=(G*A1+Z(I)*A2)/E	*	0
119	24 Z(I)=(Z(I)*A1-G*A2)/E	*	0
120 C	NORMALIZED EIGENVECTOR OBTAINED		
121 C PUT	Y(I) (REAL PART) IN EIGVECT(NUMOFEV,I)		
122	DO 100 I=1,N	*	0
123	100 EIGVEC(NUM,I)=Y(I)	*	0
124	NUM=NUM+1	*	0
125 C PUT	Z(I) (COMPLEX PART) IN EIGVECT(NUMOFEV,I)		
126	DO 101 I=1,N	*	0
127	101 EIGVEC(NUM,I)=Z(I)	*	0
128	RETURN	*	0
129	END		

* PROGRAM EVALUATOR AND TESTER REPORT *
* FOR SUBROUTINE IMVEC *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS 127

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	118	92.9
NONEXECUTABLE SOURCE	3	2.4
COMMENT	6	4.7
NONSTANDARD	0	0.0
BRANCH	17	N/A
CALL	0	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

THIS SUBROUTINE WAS NOT EXECUTED

* PROGRAM EVALUATOR AND TESTER REPORT *

* FOR PROGRAM UNSE18 *

SYNTACTIC PROFILE

TOTAL NUMBER OF SOURCE STATEMENTS

537

TYPE OF STATEMENT	NUMBER	PERCENT OF TOTAL
EXECUTABLE SOURCE	470	87.5
NONEXECUTABLE SOURCE	19	3.5
COMMENT	48	8.9
NONSTANDARD	1	.2
BRANCH	134	N/A
CALL	8	N/A
UNFORMATTED I/O	0	N/A
FORMATTED I/O	0	N/A
ASSERTION DIRECTIVES	0	N/A

OPERATIONAL PROFILE

TOTAL EXECUTION COUNT

2347

TYPE OF STATEMENT	NO EXECUTED	PERCENT EXECUTED
EXECUTABLE SOURCE	247	52.6
BRANCH	61	45.5
CALL	4	50.0
UNFORMATTED I/O	0	100.0
FORMATTED I/O	0	100.0

20E

* PROGRAM EVALUATOR AND TESTER REPORT *
* SUBROUTINE OPERATIONAL SUMMARY *

SUBROUTINE NAME	NO OF STATEMENTS EXECUTED	PERCENT EXECUTED	NO OF CALLS EXECUTED	PERCENT EXECUTED	NO OF BRANCHES EXECUTED	PERCENT EXECUTED
UNSETG	125	80.1	4	50.0	35	60.3
CALCEV	26	70.3	0	100.0	6	54.5
REALIT	28	90.3	0	100.0	5	71.4
RVEC	68	98.6	0	100.0	15	88.2
PROGRAM	247	52.6	4	50.0	61	45.5

SUBROUTINES NOT EXECUTED

COMPIT
INVEC

* PROGRAM EVALUATOR AND TESTER REPORT *
* SUBROUTINE EXECUTION SUMMARY *

SUBROUTINE NAME	NUMBER TIMES ENTERED	TOTAL EXECUTION COUNT	TOTAL CALL COUNT	PERCENT OF TOTAL EXECUTIONS	TOTAL BRANCH COUNT	PERCENT OF TOTAL EXECUTIONS	TOTAL I/O COUNT	PERCENT OF TOTAL EXECUTIONS
UNSEIG	1	560	14	2.50	121	21.61	0	0.00
CALCEV	3	34	0	0.00	7	20.59	0	0.00
REALIT	7	515	0	0.00	32	6.21	0	0.00
RVEC	4	1236	0	0.00	170	13.73	0	0.00

```

*****
* PROGRAM EVALUATOR AND TESTER REPORT *
* SUBROUTINE TIMING SUMMARY *
*****
    
```

TOTAL EXECUTION TIME FOR THIS CASE .013 SEC

TOTAL TIME IN POSTPROCESSOR 1.260 SEC

23E

ABBREVIATIONS USED IN THE FORTRAN VERIFIER SYMBOL TABLES FOLLOWING THE PROGRAM UNIT LISTINGS

TYPE KEY	USAGE KEY	ATTRIBUTE KEY
COLUMN 1.	COLUMNS 1 AND 2,	COLUMN 1.
E EXPLICITLY TYPED	FA ARITHMETIC STATEMENT FUNCTION ARGUMENT	G IN COMMON
COLUMN 2.	FN FUNCTION	COLUMN 2.
I INTEGER	E EXTERNAL (FUNCTION OR SUBROUTINE)	E IN AN EQUIVALENCE STATEMENT
R REAL	GT ASSIGNED GOTO VARIABLE	COLUMN 3.
D DOUBLE PRECISION	IF INTRINSIC FUNCTION	A DUMMY ARGUMENT
C COMPLEX	SN SUBROUTINE	COLUMN 4.
L LOGICAL	V VARIABLE	S VALUE SET BY PROGRAM UNIT
H HOLLERITH		COLUMNS 5 AND 6.
		S SCALAR
		AX ARRAY WITH X DIMENSIONS

ORIGINAL PAGE IS
OF POOR QUALITY

C FOR DESCRIPTION OF METHOD USED IN THIS PROGRAM, SEE **THE OR
C TRANSFORMATION** BY FRANCIS, COMPTON JOURNAL, VOL. 4 PP. 265,332
C START HERE TO PRODUCE UPPER HESSENBERG MATRIX

```
1  DIMENSION A(10,10),VEC(8), EIGVEC(4,4), INT(10),CQ(10),SS(10)
2  DIMENSION Z(10),D(10,10),BE(10,10)
3  DATA A / 0.1,5.8,1.9,3.2,6*0.0,2.8,3.9,0.7,0.1,6*0.0,
3  Z 6.2,3.5,0.6,1.2,6*0.0,2.9,6.2,1.7,3.8 /
```

*** MISSING DECLARATORS OR DATA-ITEMS

*** ILLEGAL PUNCTUATION IN DATA STMT

```
4  NMAX = 10
5  NN = 4
6  NVEC = 4
7  N=NN
8  K=N-2
9  DO1J=1,K
10 K1=J+1
11 K2=J+2
12 GREAT= ABS(A(K1,J))
13 MAX=K1
14 DO2I=K2,N
15 AB= ABS(A(I,J))
16 IF (GREAT,GE,AB) GO TO 2
17 3 GREAT=AB
18 MAX=I
19 2 CONTINUE
C INT(I) GIVES INFORMATION ABOUT INTERCHANGES
20 INT(J)=MAX
21 IF (MAX.EQ.K1) GO TO 5
22 4 DO6L=J,N
23 E=A(K1,L)
24 A(K1,L)=A(MAX,L)
25 6 A(MAX,L)=E
26 DO7L=1,N
27 E=A(L,K1)
28 A(L,K1)=A(L,MAX)
29 7 A(L,MAX)=E
C INTERCHANGE DONE
30 5 DO8I=K2,N
31 IF (A(I,J).EQ.0.) GO TO 8
32 A(I,J)=-A(I,J)/A(K1,J)
33 DO11M=K1,N
34 11 A(I,M)=A(I,M)+A(I,J)*A(K1,M)
35 8 CONTINUE
C LEFT MULT. DONE
36 DO 2000 M=K2,N
37 IF (A(M,J).EQ.0.) GO TO 2000
38 10 DO 2001 I=1,N
39 2001 A(I,K1)=A(I,K1)+A(I,M)*A(M,J)
40 2000 CONTINUE
41 1 CONTINUE
```

C RIGHT MULT. DONE
C HESSENBERG MATRIX NOW FORMED AS A(I,J). MULTIPLIERS STORED WITH IT

```

43 DO 105 J=1,N
44 105 D(I,J)=A(I,J)
      C PUT APPROPRIATE ZEROS WHERE MULTIPLIERS WERE STORED
45 DO 106 I=3,N
46 L=I-2
47 DO 106 J=1,L
48 106 A(I,J)=0.0
49 K1=N+1
50 R=0.0
51 C=0.0
52 DO33I=1,N
53 R=R+ABS(A(I,I))
54 33 C=C+ABS(A(I,N))
55 DO34I=2,N
56 RS=0.0
57 CS=0.0
58 L=I-1
59 DO35J=L,N
60 RS=RS+ABS(A(I,J))
61 K1J=K1-J
62 K1I=K1-I
63 35 CS=CS+ABS(A(K1J,K1I))
64 IF (RS,LE,R) GO TO 36
65 37 R=RS
66 36 IF (CS,LE,C) GO TO 34
67 44 C=CS
68 34 CONTINUE
69 IJX=1
70 IF (R,GE,C) GO TO 54
      C DELTA IS CRITERION FOR STOPPING AN ITERATION
71 53 DELTA=R+1.0E-11
      C DEL IS ARBITRARY SHIFT
72 DEL=R/17.1829
73 GO TO 153
74 54 DELTA=C+1.0E-11
75 DEL=C/17.1829
      C DELTA COMPUTED
      C PRODUCE A*PI, ITERATE ON THIS TO BREAK UP EIGENVALUES OF EQUAL MODULUS
76 153 DO 107 I=1,N
77 107 A(I,I)=A(I,I)-DEL
78 55 IF (N,NE,1) GO TO 48
      C SUBROUTINE CALCEV CALCULATES EIGENVALUES OF 1X1 OR 2X2 MATRICES.
      C PI IS ADDED BACK IN CALCEV
79 47 CALL CALCEV(A,NMAX,DEL,VEC,IJX,N,1)
80 GO TO 76
81 48 IF (N,NE,2) GO TO 80
82 49 CALL CALCEV(A,NMAX,DEL,VEC,IJX,N,2)
83 GO TO 76
      C OR TRANSFORMATION STARTS HERE
      C INITIALIZE FOR DETERMINING SUBSEQUENT SHIFTS
84 A1=0.0
85 A3=0.0
86 A4=0.0
      C RE-ENTER AFTER REALIT OR COMPT SUBROUTINES CALLED
      C SELECT FIRST (ZERO) SUB-DIAGONAL ELEMENT FROM BOTTOM
87 61 L=N
88 DO56 I=2,N
89 56 I=I+2-1
90 IF (ABS(A(N2I,42I-1)),LT,DELTA) GO TO 57
91 56 L=L-1
      C VAL E OF L SELECTED
92 57 IF (L,NE,N) GO TO 59
93 58 CALL CALCEV(A,NMAX,DEL,VEC,IJX,N,1)
94 N=N-1

```

```

96 59 IF (C.NE.0) GO TO 75
97 60 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
98 N=N-2
99 GO TO 55
C DETERMINE EIGENVALUES OF 2X2 MATRIX -- ROW 1 = A(N-1,N-1) A(N-1,N)
C ROW 2 = A(N,N-1) A(N,N)
C AND BASE SHIFT ON THESE
100 75 B=A(N-1,N-1)*A(N,N)
101 C=A(N-1,N-1)*A(N,N)+A(N,N-1)*A(N-1,N)
102 C1=A(N-1,N-1)-A(N,N)
103 DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)
104 IF (DISC.GE.0.) GO TO 63
105 62 R1=0.5*B
106 R2=.5*SQRT(AUS(DISC))
107 ANUM=(R1-AR3)*(R1-AR3)+(R2-AR4)*(R2-AR4)
108 DENOM=AR3*AR3+AR4*AR4
109 IF (DENOM.NE.0.) GO TO 200
110 201 RHO=0.0
111 SIGMA=0.0
112 GO TO 203
113 200 RHO=C
114 SIGMA=B
115 203 AR3=R1
116 AR4=R2
117 CALL COMFIT(A,NMAX,L,N,RHO,SIGMA)
118 GO TO 61
119 63 IF (B.GE.0.) GO TO 65
120 64 R1=.5*(B+SQRT(DISC))
121 GO TO 66
122 65 R1=.5*(B+SQRT(DISC))
123 IF (R1.NE.0.) GO TO 66
124 83 R2=0.0
125 GO TO 84
126 66 R2=C/R1
C ROOTS OF 2*2 MATRIX FOUND
127 84 IF (ABS(A(N,N)-R1).GE. AUS(A(N,N)-R2)) GO TO 68
128 67 AR2=R1
129 GO TO 69
130 68 AR2=R2
131 69 IF (AR1.NE.0.) GO TO 71
132 72 SHIFT=0.0
133 GO TO 73
134 71 SHIFT=AR2
135 73 AR1=AR2
136 CALL REALIT(A,NMAX,L,N,SHIFT,CC,SS)
137 GO TO 61
C ARE VECTORS NEEDED
138 76 IF (NVEC.EQ.0) GO TO 600
C INVERSE ITERATION IS USED FOR EIGENVECTORS
139 550 SMALL=DELTA*.1,CF=30
140 NUM=0
141 J1=1
142 NM=NM+NM
C C1 IS REAL PART OF EIGENVALUE
143 513 C1=VEC(J1)
C C2 IS COMPLEX PART OF EIGENVALUE
144 C2=VEC(J1+1)
145 NM=NUM+1
C REPLACE HESSENBURG MATRIX AND MULTIPLIERS
146 DO 570 I=1,NM
147 DO 570 J=1,NM
148 570 A(I,J)=D(I,J)
149 IF (C2.NE.0.) GO TO 561
150 CALL RVEC(A,NMAX,INT,C1,SMALL,NM,NUM,EIGVEC,NVEC,CC,SS)
151 561 J1=J1+1
152 IF (NVEC.EQ.NUM) GO TO 600

```

```

154      561 CALL IMVEC(A,NMAX,INT,C1,C2,SPALL,NN,NUH,EIGVEC,NVEC,CC,SS,7,RE)
      C NEXT STATEMENT IGNORES CONJUGATE COMPLEX EIGENVALUES
155      J1=J1+4
156      IF(NVEC.EQ.0)GO TO 600
157      GO TO 513
158      600 CONTINUE
159      WRITE(6,700) (VEC(I),I=1,NN)
160      DO 20 I=1,NN
161      20  WRITE(6,700) (EIGVEC(I,J),J=1,NN)
162      700  FORMAT(14,5E20.8 / 5E20.8)
163      END

```

*** P-U NOT SAVED FOR PASS2 PROCESSING

ORIGINAL PAGE IS
OF POOR QUALITY

PROGRAM UNIT *MAIN

NAME TYPE USE ATTRIBUTES REFERENCES

ABS	R	IF		12	15	53	54	60	63	90	106
				127							
AB	R	V	SS	15	16	17					
ANUM	R	V	SS	107							
AR1	R	V	SS	84	131	135					
AR2	R	V	SS	128	130	134	135				
AR3	R	V	SS	65	107	108	115				
AR4	R	V	SS	86	107	108	116				
A	R	V	SA2	1	3	12	15	23	24	25	27
				28	29	31	32	34	37	39	44
				48	53	54	60	63	77	79	82
				90	93	97	100	101	102	103	117
				127	136	148	150	154			
BB	R	V	A2	2	154						
B	R	V	SS	100	105	114	119	120	122		
CALCEV		SN		79	82	83	97				
CC	H	V	A1	1	136	150	154				
COMPI		SN		117							
CS	R	V	SS	57	63	66	67				
C1	R	V	SS	102	103	143	150	154			
C2	R	V	SS	144	149	154					
C	R	V	SS	51	54	66	67	70	74	75	101
				113	126						
DELTA	R	V	SS	71	74	90	139				
DEL	R	V	SS	72	75	77	79	82	93	97	
DENDM	R	V	SS	160	109						
DISC	R	V	SS	103	104	106	120	122			
D	R	V	SA2	2	44	148					
EIGVEC	R	V	A2	1	150	154	161				
E	R	V	SS	23	25	27	29				
GREAY	R	V	SS	12	16	17					
IJK	I	V	SS	69	79	82	93	97			
INVFC		SN		154							
INT	I	V	SA1	1	20	150	154				
I	I	V	SS	14	15	18	30	31	32	34	38
				39	42	44	45	46	48	52	53
				54	55	58	60	62	76	77	88
				69	146	148	159	160	161		
J1	I	V	SS	141	143	144	151	155			
J	I	V	SS	9	10	11	12	15	20	22	31
				32	34	37	39	43	44	47	48
				59	60	61	147	148	161		
K11	I	V	SS	62	63						
K1J	I	V	SS	61	63						
K1	I	V	SS	10	12	13	21	23	24	27	28
				32	33	34	39	49	61	62	
K2	I	V	SS	11	14	30	36				
K	I	V	SS	8	9						
L	I	V	SS	22	23	24	25	26	27	28	29
				46	47	58	59	87	91	92	96
				117	136						
MAX	I	V	SS	13	18	20	21	24	25	28	29
H	I	V	SS	33	34	36	37	39			
NMAX	I	V	SS	4	79	82	93	97	117	136	150
				154							
NN	I	V	SS	5	7	142	146	147	150	154	159

NUM	1	V	SS	140	145	150	152	154	156
NVEC	1	V	SS	6	138	150	152	154	156
N21	1	V	SS	82	90				
M	1	V	SS	7	8	14	22	26	30
				38	42	43	45	49	52
				59	76	78	79	81	82
				89	92	93	94	96	97
				101	102	103	117	127	136
REALIT		SN		136					
RHO	R	V	SS	110	113	117			
RS	R	V	SS	56	60	64	65		
RVEC		SN		150					
R1	R	V	SS	105	107	115	120	122	123
				128					
R2	R	V	SS	156	107	116	124	126	127
R	R	V	SS	50	53	64	65	70	71
SHIFT	R	V	SS	132	134	136			
SIGMA	R	V	SS	111	114	117			
SMALL	R	V	SS	139	150	154			
SORY	RR	FN		106	120	122			
SS	R	V	A1	1	136	150	154		
VEC	R	V	A1	1	79	82	93	97	143
Z	R	V	A1	2	154				
105				42	43	44			
106				45	47	48			
107				76	77				
10				38					
11				33	34				
153				73	76				
1				9	41				
2000				36	37	40			
2001				38	39				
200				109	113				
201				110					
203				112	115				
20				160	161				
2				14	16	19			
33				52	54				
34				55	66	68			
35				59	63				
36				64	66				
37				65					
3				17					
44				67					
47				79					
48				78	81	95			
49				82					
4				22					
513				143	153	157			
53				71					
54				70	74				
550				139					
55				78	99				
561				149	154				
56				88	91				
570				146	147	148			
57				90	92				
58				93					
59				92	96				
5				21	30				
600				138	152	156	158		
60				97					
61				87	118	137			
62				105					

64	120	122	
65	119	123	126
66	121		
67	128		
68	127	130	
69	129	131	
6	22	25	
700	159	161	162
71	131	134	
72	132		
73	133	135	
75	96	190	
76	40	83	138
7	26	29	
80	81	84	
83	124		
84	125	127	
8	30	31	35

*** ILLEGAL LAST EXECUTABLE STMT

```
1 SUBROUTINE CALCEV(A,NMAX,DEL,VEC,IJK,N,H)
2 DIMENSION A(NMAX,NMAX),VEC(1)
3 IF (H,NE.1) GO TO 2
4 1 A1=0.0
5 A(N,N)=A(N,N)+DEL
6 C STCPH EIGENVALUES IN VEC
7 VEC(IJK)=A(N,N)
8 VEC(IJK+1)=A1
9 IJK=IJK+2
10 RETURN
11 2 B=A(N+1,N+1)+A(N,N)
12 C=A(N+1,N+1)*A(N,N)+A(N,N+1)*A(N+1,N)
13 C1=A(N+1,N+1)-A(N,N)
14 DISC=C1+C1+4.0*A(N,N+1)*A(N+1,N)
15 IF (DISC.LT.0.) GO TO 9
16 4 A1=0.0
17 IF (H,GE.0.) GO TO 6
18 R1=0.5*(H+SQRT(DISC))
19 GO TO 7
20 6 R1=0.5*(H-SQRT(DISC))
21 IF (R1,NE.0.) GO TO 7
22 10 R2=0.0
23 GO TO 11
24 7 R2=C/R1
25 11 R1=R1+DEL
26 R2=R2+DEL
27 VEC(IJK)=R2
28 VEC(IJK+1)=A1
29 VEC(IJK+2)=R1
30 VEC(IJK+3)=A1
31 GO TO 8
32 9 R1=0.5*B
33 A1=-.5*SQRT(ABS(DISC))
34 R1=R1+DEL
35 VEC(IJK)=R1
36 VEC(IJK+1)=-A1
37 VEC(IJK+2)=R1
38 VEC(IJK+3)=A1
39 8 IJK=IJK+4
40 RETURN
41 END
```

ORIGINAL PAGE IS
OF POOR QUALITY

PROGRAM UNIT CALDEV

ARGUMENTS A NMAX DEL VEC IJK N M

NAME TYPE USE ATTRIBUTES REFERENCES

ABS	R	IF		32							
AI	R	V	SS	4	7	15	27	29	32	35	37
A	R	V	ASA2	1	2	5	6	10	11	12	13
B	R	V	SS	10	16	17	19	31			
C1	R	V	SS	12	13						
C	R	V	SS	11	23						
DEL	R	V	A S	1	5	24	25	33			
DISC	R	V	SS	13	14	17	19	32			
IJK	I	V	ASS	1	6	7	8	26	27	28	29
				34	35	36	37	38			
M	I	V	A S	1	3						
NMAX	I	V	A S	1	2						
N	I	V	A S	1	5	6	10	11	12	13	
R1	R	V	SS	17	19	20	23	24	28	31	33
				34	36						
R2	R	V	SS	21	23	25	26				
SORT	ER	FN		17	19	32					
VEC	R	V	ASA1	1	2	6	7	26	27	28	29
				34	35	36	37				
10				21							
11				22	24						
1				4							
2				3	10						
4				15							
6				16	19						
7				18	20	23					
8				30	38						
9				14	31						

```
1 SUBROUTINE REALTIA(NMAX,L,N,SHIFT,C,S)
2 DIMENSION A(NMAX,NMAX),C(1),S(1)
3 IF (SHIFT,EQ,0.) GO TO 2
4 DO3 I=L,N
5   1 A(I,1)*A(I,1)-SHIFT
6   2 M=N-1
7   DO4 I=L,M
8     DENOM= SQRT(A(I,1)**2+A(I+1,1)**2)
9     IF (DENOM,GT,0.) GO TO 10
10    C(1)=1.0
11    S(1)=0.0
12    GO TO 4
13    10 C(1)=A(I,1)/DENOM
14    S(1)=A(I+1,1)/DENOM
15    A(I,1)=DENOM
16    12 I=I+1
17    DO 35 J=I+1,N
18      F=A(I,J)
19      A(I,J)=A(I+1,J)*S(1)+F*C(1)
20      A(I+1,J)=A(I+1,J)*C(1)-F*S(1)
21    35 CONTINUE
22    4 CONTINUE
C    TRIANGULAR MATRIX FORMED
23    DO5 J=L,M
24      DO6 I=L,J
25        F=A(I,J)
26        A(I,J)=A(I,J+1)*S(J)+F*C(J)
27        6 A(I,J+1)=A(I,J+1)*C(J)-F*S(J)
28        A(J+1,J)=A(J+1,J+1)*S(J)
29        5 A(J+1,J+1)=A(J+1,J+1)*C(J)
C    TRANSFORMATION COMPLETED
30    IF (SHIFT,EQ,0.) RETURN
31    7 DO9 I=L,N
32      9 A(I,1)=A(I,1)+SHIFT
33    8 RETURN
34    END
```

PROGRAM UNIT REALIT

ARGUMENTS A NHAX L N SHIFT C S

NAME TYPE USE ATTRIBUTES REFERENCES

A	R	V	ASA2	1	2	5	8	13	14	15	18
				19	20	25	26	27	28	29	32
C	R	V	ASA1	1	2	10	13	19	20	26	27
				29							
DENOM	R	V	SS	8	9	13	14	15			
F	R	V	SS	18	19	20	25	26	27		
I1	I	V	SS	16	17						
I	I	V	SS	4	5	7	8	10	11	13	14
				15	16	18	19	20	24	25	26
				27	31	32					
J	I	V	SS	17	18	19	20	23	24	25	26
				27	28	29					
L	I	V	A S	1	4	7	23	24	31		
M	I	V	SS	6	7	23					
NHAX	I	V	A S	1	2						
N	I	V	A S	1	4	6	17	31			
SHIFT	H	V	A S	1	3	5	30	32			
SHORT	ER	FN		8							
S	R	V	ASA1	1	2	11	14	19	20	26	27
				28							
10				9	13						
11				10							
12				16							
1				4							
2				3	6						
35				17	21						
3				4	5						
4				7	12	22					
5				23	29						
6				24	27						
7				31							
8				33							
9				31	32						

```

1  SUBROUTINE COMPT(A,NMAX,L,N,RHO,SIGMA)
2  DIMENSION A(NMAX,NMAX)
3  C HOUSEHOLDER TYPE MATRICES USED INSTEAD OF ROTATIONS(SEE FRANCIS PAPER)
4  M=N-1
5  DO 1 I=L,N
6  IF (I,NE,L) GO TO 3
7  B1=A(L,I)*A(L,I)+SIGMA*A(L,I+1)*A(I+1,I)+RHO
8  B2=A(L+1,I)*A(L,I)+A(L+2,I+1)*SIGMA
9  B3=A(L+1,I)*A(L+2,I+1)
10 A(L+2,I)=0.0
11 GO TO 6
12 3 B1=A(I,I-1)
13 B2=A(I-1,I-1)
14 IF (I,EQ,M) GO TO 5
15 4 B3=A(I+2,I-1)
16 GO TO 6
17 5 B3=0.0
18 D=SQRT(B1*B1+B2*B2+B3*B3)
19 IF (D,GE,0.) GO TO 8
20 7 D=-D
21 IF (D,EQ,0.) GO TO 1
22 9 C1=B2/(D+B1)
23 C2=B3/(D+B1)
24 E=2.0/(1.0+C1*C1+C2*C2)
25 DO 10 K=I,N
26 IF (I,EQ,M) GO TO 12
27 11 G=E*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))
28 A(I,K)=A(I,K)-G
29 A(I+1,K)=A(I+1,K)+C1*G
30 A(I+2,K)=A(I+2,K)+C2*G
31 GO TO 10
32 12 G=E*(A(I,K)+C1*A(I+1,K))
33 A(I,K)=A(I,K)-G
34 A(I+1,K)=A(I+1,K)+C1*G
35 10 CONTINUE
36 IF (I,EQ,L) GO TO 14
37 13 A(I,I-1)=D
38 A(I+1,I-1)=0.0
39 IF (I,EQ,M) GO TO 14
40 15 A(I+2,I-1)=0.0
41 C ROW OPERATION COMPLETED
42 14 J=I+2
43 IF (J,LE,N) GO TO 17
44 J=N
45 DO 18 K=L,J
46 IF (I,EQ,M) GO TO 20
47 19 G=E*(A(K,I)+C1*A(K,I+1)+C2*A(K,I+2))
48 A(K,I)=A(K,I)-G
49 A(K,I+1)=A(K,I+1)+C1*G
50 A(K,I+2)=A(K,I+2)+C2*G
51 GO TO 18
52 20 G=E*(A(K,I)+C1*A(K,I+1))
53 A(K,I)=A(K,I)-G
54 A(K,I+1)=A(K,I+1)+C1*G
55 18 CONTINUE
56 J=J+3
57 IF (J,GT,N) GO TO 1
58 21 G=E*C2*A(I+3,I+2)
59 A(I+3,I)=G
60 A(I+3,I+1)=C1*G

```

ORIGINAL PAGE IS
OF POOR QUALITY



14E

PROGRAM UNIT COMPI

ARGUMENTS A NMAX L N RHO SIGMA

NAME TYPE USE ATTRIBUTES REFERENCES

A	R	V	ASA2	1	2	6	7	8	9	11	12
				14	26	27	28	29	31	32	33
				36	37	39	45	46	47	48	50
B1	R	V	SS	51	52	56	57	58	59		
B2	R	V	SS	6	11	17	18	21	22		
B3	R	V	SS	7	12	17	21				
C1	R	V	SS	14	16	17	22				
				21	23	26	28	31	33	43	47
				50	52	58					
C2	R	V	SS	22	23	26	29	45	48	56	59
D	R	V	SS	17	19	20	21	22	36		
E	R	V	SS	23	26	31	45	50	56		
G	R	V	SS	26	27	28	29	31	32	33	45
				46	47	48	50	51	52	56	57
				58	59						
I	I	V	SS	4	5	11	12	13	14	24	25
				26	27	28	29	31	32	33	35
				36	37	38	39	40	44	48	46
				47	48	50	51	52	54	56	57
				58	59						
J	I	V	SS	40	41	42	43	54	55		
K	I	V	SS	24	26	27	28	29	31	32	33
				43	45	46	47	48	50	51	52
L	I	V	A S	1	4	5	6	7	8	9	35
				43							
M	I	V	SS	3	4	13	25	38	44		
NMAX	I	V	A S	1	2						
N	I	V	A S	1	3	24	41	42	55		
RHO	R	V	A S	1	6						
SIGMA	R	V	A S	1	6	7					
SORT	ER	FN		17							
10				24	30	34					
11				26							
12				25	31						
13				36							
14				35	38	40					
15				39							
16				42							
17				41	43						
18				43	49	53					
19				45							
1				4	20	55	60				
20				44	50						
21				56							
2				6							
3				5	11						
4				14							
5				13	16						
6				10	15	17					
7				19							
8				18	20						
9				21							



```

1  SUBROUTINE RVEC(A,NMAX,INT,C1,SMALL,N,NUM,EIGVEC,NVEC,INTER,Y)
2  DIMENSION A(NMAX,NMAX),INTER(1),Y(1),INT(1),EIGVEC(NMAX,NMAX)
3  DO1 I=1,N
4  1 A(I,1)=A(I,1)-C1
5  DO2 I=2,N
6  IF( ABS(A(I-1,I-1)),GE, ABS(A(I,I-1)))GO TO 4
7  3 INTER(I)=I
8  J=I-1
9  DO5 K=J,N
10 B=A(I-1,K)
11 A(I-1,K)=A(I,K)
12 5 A(I,K)=B
13 GO TO 6
14 4 INTER(I)=I-1
15 6 IF (A(I,I-1),EQ,0.) GO TO 2
16 C=A(I,I-1)/A(I-1,I-1)
17 A(I,I-1)=C
18 DO 8 K=I,N
19 8 A(I,K)=A(I,K)+C*A(I-1,K)
20 2 CONTINUE
C TRIANGULAR MATRIX FORMED
21 DO9 I=1,N
22 IF( ABS(A(I,1)),GE,SMALL)GO TO 9
23 10 A(I,1)=SMALL
24 9 CONTINUE
C NORMALISED ((EIGENVECTOR)) OF HESSENBURG MATRIX OBTAINED FOR RHS = 1
25 DO 5555 I=1,N
26 5555 Y(I)=1.0
27 DO 8888 JJJ,1,4
28 7777 Y(N)=Y(N)/A(N,N)
29 B=Y(N)
30 DO11 I=2,N
31 E=0.0
32 M=N+1-I
33 DO12 J=2,I
34 L=N+J-I
35 12 E=E+Y(L)*A(M,L)
36 Y(M)=(Y(M)-E)/A(M,M)
37 IF( ABS(B),GE, ABS(Y(M)))GO TO 11
38 B=Y(M)
39 11 CONTINUE
40 DO14 I=1,N
41 14 Y(I)=Y(I)/B
42 IF(JJJJ,EQ,4)GO TO 6666
43 DO15 I=2,N
44 IF (INTER(I),NE,1) GO TO 15
45 16 B=Y(I)
46 Y(I)=Y(I-1)
47 Y(I-1)=B
48 15 Y(I)=Y(I)+A(I,I-1)*Y(I-1)
49 8888 CONTINUE
C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED (NOT NORMALIZED)
50 6666 M=N-1
51 DO21 I=2,M
52 J=N+1-I
53 L=J+1
54 DO20 K=L,N
55 20 Y(K)=Y(K)-Y(J)*A(K,J+1)
56 PRINT *,Y

```

*** ILLEGAL SYNTAX OF SUBSCRIPT

57 IF (J,Eq,LL) GO TO 21
58 19 B=Y(J)
59 Y(J)*Y(LL)
60 Y(LL)=B
61 21 CONTINUE
C UNNORMALIZED EIGENVECTOR OF ORIGINAL MATRIX OBTAINED
62 B=0.0
63 DO22 I=1,N
64 IF (ABS(B).GE. ABS(Y(I)))GC TC 22
65 23 B=Y(I)
66 22 CONTINUE
67 DO24 I=1,N
68 24 Y(I)*Y(I)/B
C NORMALIZED EIGENVECTOR OBTAINED
C PUT Y(I) (REAL EIGENVECTOR) IN EIGVECT(NLMOFEV,I)
69 DO 100 I=1,N
70 100 EIGVEC(NUM,I)=Y(I)
71 RETURN
72 END

ORIGINAL PAGE IS
OF POOR QUALITY

PROGRAM UNIT

RVEC

ARGUMENTS

A

NMAX

INT

C1

SHALL

N

NUM

E1GVEC

NVEC

INTER Y

NAME

TYPE

USE ATTRIBUTES REFERENCES

ABS	R	IF		6	22	37	64				
A	R	V	ASA2	1	2	4	6	10	11	12	15
				16	17	19	22	23	28	33	36
				48	55						
B	R	V	SS	10	12	29	37	38	41	45	47
				58	60	62	64	65	68		
C1	R	V	A S	1	4						
C	R	V	SS	16	17	19					
E1GVEC	R	V	ASA2	1	2	70					
E	R	V	SS	31	35	36					
INTER	I	V	ASA1	1	2	7	14	44			
INT	I	V	A A1	1	2	56					
I	I	V	SS	3	4	5	6	7	8	10	11
				12	14	15	16	17	18	19	21
				22	23	25	26	30	32	33	34
				40	41	43	44	45	46	47	48
				51	52	63	64	65	67	68	69
				70							
JJJJ	I	V	SS	27	42						
J	I	V	SS	8	9	33	34	52	53	55	57
				58	59						
K	I	V	SS	9	10	11	12	18	19	54	55
LL	I	V	SS	56	57	59	60				
L	I	V	SS	34	35	53	54				
M	I	V	SS	32	35	36	37	38	50	51	
NMAX	I	V	A S	1	2						
NUM	I	V	A S	1	70						
NVEC	I	V	A S	1							
N	I	V	A S	1	3	5	9	18	21	25	28
				29	30	32	34	40	43	50	52
				54	56	63	67	69			
SHALL	R	V	A S	1	22	23					
Y	R	V	ASA1	1	2	26	28	29	35	36	37
				38	41	45	46	47	48	55	58
				59	60	64	65	68	70		
				69	70						
100				23							
10				30	37	39					
11				33	35						
12				38							
13				40	41						
14				43	44	48					
15				45							
16				50							
19				3	4						
1				54	55						
20				51	57	61					
21				63	64	66					
22				65							
23				67	68						
24				5	15	20					
3				7							
4				6	14						

5
6666
6
7777
8888
8
9

25
9
42
13
28
27
18
21

20
12
50
15
49
19
22

24

ORIGINAL PAGE IS
OF POOR QUALITY

```

1      SUBROUTINE IHVEC(A,NMAX,INT,C1,AMU,SMALL,N,NUM,EIGVEC,NVEC,INTER,
2      1Y,Z,B)
3      DIMENSION A(NMAX,NMAX),INTER(1),Y(1),Z(1),INT(1),B(NMAX,NMAX)
4      EIGVEC(NMAX,NMAX)
5      DO 50 I=1,N
6      DO 50 J=1,N
7      B(I,J)=0.0
8      DO 1 I=1,N
9      A(I,I)=A(I,I)-C1
10     B(I,I)=B(I,I)-AMU
11     DO 2 I=2,N
12     C=A(I-1,I-1)+A(I-1,I)+B(I-1,I-1)+B(I-1,I)
13     D=A(I,I-1)+A(I,I-1)+B(I,I-1)+B(I,I)
14     IF (C,GE,D) GO TO 4
15     3 INTER(I)=I
16     G=C
17     C=D
18     D=G
19     J=I-1
20     DO 5 K=J,N
21     G=A(I-1,K)
22     A(I-1,K)=A(I,K)
23     A(I,K)=G
24     G=B(I-1,K)
25     B(I-1,K)=B(I,K)
26     B(I,K)=G
27     GO TO 6
28     4 INTER(I)=I-1
29     IF (D,EQ,0.0) GO TO 2
30     G=A(I,I-1)
31     A(I,I-1)=(G+A(I-1,I-1)+B(I-1,I-1)+B(I-1,I))/C
32     B(I,I-1)=(G+A(I-1,I-1)+B(I-1,I-1)+B(I-1,I))/C
33     DO 6 K=1,N
34     A(I,K)=A(I,K)+A(I,I-1)*A(I-1,K)-B(I,I-1)*B(I-1,K)
35     B(I,K)=B(I,K)+B(I,I-1)*A(I-1,K)+A(I,I-1)*B(I-1,K)
36     2 CONTINUE
37     C TRIANGULAR MATRIX FORMED
38     DO 9 I=1,N
39     C=A(I,I)*A(I,I)+B(I,I)*B(I,I)
40     IF (C,GE,SMALL) GO TO 9
41     A(I,I)=SORT(SMALL)
42     B(I,I)=0.0
43     9 CONTINUE
44     DO 1111 I=1,N
45     Y(I)=1.0
46     Z(I)=0.0
47     1111 DO 8888 JJJ=1,4
48     C=A(N,N)*A(N,N)+B(N,N)*B(N,N)
49     G=Y(N)
50     Y(N)=(G+A(N,N)+Z(N)*B(N,N))/C
51     Z(N)=(Z(N)+A(N,N)+G*B(N,N))/C
52     DO 17 I=2,N
53     E=0.0
54     F=0.0
55     M=N+1-I
56     DO 18 J=2,I
57     L=N+J-1
58     F=E+Y(L)*A(M,I-1)+Z(L)*B(M,I)
59     D=A(M,M)*A(M,M)+B(M,M)*B(M,M)

```

```

58      Y(I)*((G-E)*A(I,M)+(Z(I)*F)*B(I,M))/D
59      Z(I)*((Z(I)-F)*A(I,M)+(G-E)*B(I,M))/D
60      E=C.0
61      DO51 I=1,N
62      DO51 I=1,N
63      F=Y(I)*Y(I)+Z(I)*Z(I)
64      IF (E,GE,F) GO TO 51
65      52 E=F
66      L=I
67      51 CONTINUE
68      A1=Y(L)
69      A2=Z(L)
70      DO14 I=1,N
71      G=Y(I)
72      Y(I)=(G*A1+7(I)*A2)/E
73      14 Z(I)=(Z(I)*A1-G*A2)/E
74      IF(JJJ,EQ,4) GO TO 6666
75      DO15 I=2,N
76      IF (INTER(I),NE,1) GO TO 30
77      16 G=Y(I)
78      Y(I)=Y(I-1)
79      Y(I-1)=G
80      G=Z(I)
81      Z(I)=Z(I-1)
82      Z(I-1)=G
83      30 Y(I)=Y(I)+A(I,I-1)*Y(I-1)+B(I,I-1)*Z(I-1)
84      15 Z(I)=Z(I)+B(I,I-1)*Y(I-1)+A(I,I-1)*Z(I-1)
85      8888 CONTINUE
86      C EIGENVECTOR OF HESSENBERG MATRIX OBTAINED
87      6666 M=M-1
88      DO21 I=2,M
89      J=M+1-I
90      L=J+1
91      DO20 K=1,N
92      Y(K)=Y(K)+Y(J)*A(K,J+1)+Z(J)*B(K,J+1)
93      20 Z(K)=Z(K)+Z(J)*A(K,J+1)+Y(J)*B(K,J+1)
94      LL=INT(M-1)

```

ORIGINAL PAGE IS
OF POOR QUALITY

*** ILLEGAL SYNTAX OF SUBSCRIPT

```

94      IF (J,EQ,LL) GO TO 21
95      19 G=Y(J)
96      Y(J)=Y(LL)
97      Y(LL)=G
98      G=Z(J)
99      Z(J)=Z(LL)
100     Z(LL)=G
101     21 CONTINUE
102     C UNNORMALIZED EIGENVECTOR OBTAINED
103     E=0.0
104     DO22 I=1,N
105     F=Y(I)*Y(I)+Z(I)*Z(I)
106     IF (E,GE,F) GO TO 22
107     23 E=F
108     L=I
109     22 CONTINUE
110     A1=Y(L)
111     A2=Z(L)
112     DO24 I=1,N
113     G=Y(I)
114     Y(I)=(G*A1+7(I)*A2)/E
115     24 Z(I)=(Z(I)*A1-G*A2)/E
116     C NORMALIZED EIGENVECTOR OBTAINED

```

```
115      DO 100 I=1,N
116      EIGVEC(NUM,I)=Y(I)
117      NUM=NUM+1
118      C PUT Z(I) (COMPLEX PART) IN EIGVEC(NUMOFFV,I)
119      DO 101 I=1,N
120      EIGVEC(NUM,I)=Z(I)
121      RETURN
122      END
```


PROGRAM UNIT INVEC

ARGUMENTS A NHAX INT C1 AHU SHALL N NUH EIGVEC
NVEC INTER Y Z B

NAME TYPE USE ATTRIBUTES REFERENCES

AHU	R	V	A S	1	8						
A1	R	V	SS	68	72	73	109	113	114		
A2	R	V	SS	69	72	73	110	113	114		
A	R	V	ASA2	1	2	7	10	11	19	20	21
				28	29	30	32	33	36	38	45
				47	48	55	56	57	59	60	83
				84	91	92					
B	R	V	ASA2	1	2	5	8	10	11	22	23
				24	29	30	32	33	36	39	45
				47	48	55	56	57	59	60	83
				84	91	92					
C1	R	V	A S	1	7						
C	R	V	SS	10	12	14	15	29	30	36	37
				45	47	48					
D	R	V	SS	11	12	15	16	27	57	59	60
EIGVEC	R	V	ASA2	1	2	116	119				
E	R	V	SS	50	55	59	60	61	64	65	72
				73	102	105	106	113	114		
F	R	V	SS	51	56	59	60	63	64	65	104
				105	106						
G	R	V	SS	14	16	19	21	22	24	28	29
				30	46	47	48	58	59	60	71
				72	73	77	79	80	82	95	97
				98	100	112	113	114			
INTER	I	V	ASA1	1	2	13	26	74			
INT	I	V	A A1	1	2	93					
I	I	V	SS	3	5	6	7	8	9	10	11
				13	17	19	20	21	22	23	24
				26	28	29	30	31	32	33	35
				36	38	39	41	42	43	49	52
				53	54	62	63	66	70	71	72
				73	75	76	77	78	79	80	81
				82	83	84	87	88	103	104	107
				111	112	113	114	115	116	118	119
JJJ	I	V	SS	44	74						
J	I	V	SS	4	5	17	18	53	54	88	89
				91	92	94	95	96	98	99	
K	I	V	SS	18	19	20	21	22	23	24	31
				32	33	90	91	92			
LL	I	V	SS	93	94	96	97	99	100		
L	I	V	SS	54	55	56	66	68	69	89	90
				107	109	110					
M	I	V	SS	52	55	56	57	58	59	60	86
				87							
NHAX	I	V	A S	1	2						
NUH	I	V	ASS	1	116	117	119				
NVEC	I	V	A S	1							
N	I	V	A S	1	3	4	6	9	18	31	35
				41	45	46	47	48	49	52	54
				62	70	75	86	88	90	93	103
				111	115	118					
SMALL	R	V	A S	1	37	38					

		ASA1	1	2	42	48	47	55
			59	63	68	71	72	77
			83	84	91	92	95	96
			109	112	113	116		97
								104
Z	R	V	ASA1	1	2	43	47	48
				60	63	69	72	73
				83	84	91	92	98
				110	113	114	119	55
				115	116			56
				118	119			59
100				41	43			80
101				70	73			81
1111				75	84			82
14				77				100
15				49	60			104
16				53	56			
17				95				
18				6	8			
19				90	92			
1				27	94	101		
20				103	105	108		
21				106				
22				111	114			
23				9	27	14		
24				76	83			
2				13				
30				12	26			
3				3	4	5		
4				62	64	67		
50				65				
51				18	24			
52				74	86			
5				25	27			
6666				28				
6				44	85			
7				31	33			
8888				35	37	40		
8								
9								

*** MISSING MAIN PGM

CALCEV

ARGUMENT ATTRIBUTES	=RSA	=I-S	=R-S	=RSA	=ISS	=I-S
CALLS SUBPROGRAMS	=I-S SORT					

COMPIT

ARGUMENT ATTRIBUTES	=RSA	=I-S	=I-S	=I-S	=R-S	=R-S
CALLS SUBPROGRAMS	SORT					

IMVEC

ARGUMENT ATTRIBUTES	=RSA	=I-S	=I-A	=R-S	=R-S	=R-S
	=I-S	=ISS	=RSA	=I-S	=ISA	=RSA
CALLS SUBPROGRAMS	=RSA SORT	=RSA				

REALIT

ARGUMENT ATTRIBUTES	=RSA	=I-S	=I-S	=I-S	=R-S	=RSA
CALLS SUBPROGRAMS	=RSA SORT					

RVEC

ARGUMENT ATTRIBUTES	=RSA	=I-S	=I-A	=R-S	=R-S	=I-S
	=I-S	=RSA	=I-S	=ISA	=RSA	

SORT

ARGUMENT ATTRIBUTES CALLED BY SUBPROGRAMS	=R-S IMVEC	COMPIT	REALIT	CALCEV
--	---------------	--------	--------	--------

1

```
00001 IDENTIFICATION DIVISION.
00002 PROGRAM-ID, AVSIM.
00003 ENVIRONMENT DIVISION.
00004 INPUT-OUTPUT SECTION.
00005 FILE-CONTROL.
00006     SELECT CDB      ASSIGN TO UT-S=SYS001.
00007     SELECT AVS      ASSIGN TO UT-S=SYS002.
00008     SELECT RPT      ASSIGN TO UT-S=SYS004.
00009 DATA DIVISION.
00010 FILE SECTION.
00011 FD CDB.
00012     01 CDBR          PIC X(80).
00013 FD AVS.
00014     01 AVSR          PIC X(04).
00015         10 FILLER    PIC X(06).
00016         10 AVSR-1     PIC X(02).
00017         10 FILLER    PIC X(6A).
00018         10 AVSR-2     PIC X(6A).
00019 FD RPT.
00020     01 RPTR          PIC X(80).
00021         05 RPTR80     PIC X(53).
00022         05 RPTR53
00023 WORKING-STORAGE SECTION.
00024     77 CTRL          PIC X(01).
00025     77 CNTR1         PIC 9(06) VALUE 99.
00026     77 CNTR2         PIC 9(03) VALUE ZEROS.
00027     77 CNTR3         PIC 9(05) VALUE ZEROS.
00028     01 SUMM-FLAG     PIC X(01) VALUE SPACE.
00029     01 RPTWS.
00030         05 FILLER     PIC X(01).
00031         05 LINE-NO     PIC X(05) VALUE SPACES.
00032         05 FILLER     PIC X(01) VALUE SPACES.
00033         05 RPTWS-1     PIC X(01).
00034         05 COBX.
00035             07 COBX-3.
00036                 10 COBX-1     PIC X(01).
00037                 10 COBX-2     PIC X(01).
00038                 10 FILLER     PIC X(01).
00039             07 COBX-77.
00040                 10 FILLER     PIC X(76).
00041                 10 COBX-80     PIC X(01).
00042         05 AVSX.
00043             10 AVSX-1     PIC X(06).
00044             10 FILLER     PIC X(03).
00045             10 AVSX-2     PIC X(71).
00046     01 RPTWSX REDEFINES RPTWS.
00047         05 RPTWSX80     PIC X(80).
00048         05 FILLER       PIC X(06).
00049         05 RPTWSX53     PIC X(53).
```

ORIGINAL PAGE IS
OF POOR QUALITY

```

00050      05 FILLER                                PIC X(29),
00051      01 HDR1,
00052      05 FILLER                                PIC X(72)  VALUE SPACES,
00053      05 FILLER                                PIC X(29)  VALUE
00054      DATE 01/31/75                             PAGE 1,
00055      05 PGE                                    PIC ZZ99,
00056      01 HDR2,
00057      05 FILLER                                PIC X(82)  VALUE SPACES,
00058      05 FILLER                                PIC X(09)  VALUE 'EXECUTION',
00059      01 HDR3,
00060      05 FILLER                                PIC X(09)  VALUE SPACES,
00061      05 FILLER                                PIC X(48)  VALUE
00062      '          ANNOTATED PROGRAM LISTING          '
00063      05 FILLER                                PIC X(41)  VALUE
00064      '          COUNT          '
00065      05 FILLER                                PIC X(23)  VALUE
00066      'SPECIFIC EXECUTION DATA'
00067      01 BLINE                                PIC X(01),
00068      PROCEDURE DIVISION,
00069      OPEN INPUT COB AVS,
00070      OPEN OUTPUT RPT,
00071      WRITE RPTR FROM BLINE AFTER POSITIONING 0 LINES,
00072      MOVE SPACES TO CTRL,
00073      MOVE 'X' TO CTRL,
00074      MC10,
00075      READ COB INTO COBX AT END GO TO MC20,
00076      READ AVS AT END GO TO MC20,
00077      IF CTRL = 'X' AND SUMM-FLAG = SPACE
00078      MOVE SPACE TO AVSX
00079      MOVE AVSR=1 TO AVSX=1
00080      MOVE AVSR=2 TO AVSX=2
00081      ELSE
00082      MOVE AVSR TO AVSX,
00083      MOVE SPACES TO RPTWS=1,
00084      IF COBX=1 = 'C' MOVE SPACE TO COBX=1,
00085      IF COBX=1 = 'X' MOVE 'C' TO COBX=1,
00086      IF COBX=1 = 'I'
00087      MOVE '1' TO SUMM-FLAG
00088      MOVE SPACE TO COBX=1
00089      ADD 01 TO CNTR2
00090      MOVE CNTR2 TO PGE
00091      WRITE RPTR FROM HDR1 AFTER POSITIONING 0 LINES
00092      WRITE RPTR FROM RLINE AFTER POSITIONING 1 LINES
00093      MOVE 00 TO CNTR1
00094      GO TO MC10,
00095      IF COBX=1 = 'E'
00096      MOVE '1' TO SUMM-FLAG
00097      MOVE SPACE TO COBX=1
00098      ADD 01 TO CNTR2
00099      MOVE CNTR2 TO PGE
00100      WRITE RPTR FROM HDR1 AFTER POSITIONING 0 LINES
00101      WRITE RPTR FROM HDR2 AFTER POSITIONING 1 LINES

```

```

00102      WRITE RPTH FROM RLINE AFTER POSITIONING 1 LINES
00103      MOVE 00 TO CNTR1
00104      MOVE 00 TO CNTR3
00105      GO TO MC10.
00106      IF CNTR1 GREATER THAN 52
00107          ADD 01 TO CNTR2
00108      MOVE CNTR2 TO PGE
00109      WRITE RPTH FROM HDR1 AFTER POSITIONING 0 LINES
00110      WRITE RPTH FROM HDR2 AFTER POSITIONING 2 LINES
00111      WRITE RPTH FROM HDR3 AFTER POSITIONING 1 LINES
00112      WRITE RPTH FROM RLINE AFTER POSITIONING 1 LINES
00113      MOVE 00 TO CNTR1.
00114      AND 01 TO CNTR1.
00115      IF COBX=1 * 1K1
00116          MOVE SPACE TO COBX=1
00117          MOVE SPACE TO LINE=NO
00118          ELSE
00119          ADD 001 TO CNTR3
00120      MOVE CNTR3 TO LINE=NO.
00121      IF SUMM=FLAG * 111
00122          MOVE SPACE TO LINE=NO
00123      WRITE RPTH FROM RPTWS AFTER POSITIONING 1 LINES
00124      GO TO MC10.
00125      IF CTRL * 1X1 GO TO MC15.
00126      IF COBX=2 NOT * SPACES AND SUMM=FLAG NOT * 111
00127          MOVE COBX=3 TO LINE=NO
00128          MOVE SPACES TO COBX=3
00129          ELSE
00130          MOVE SPACES TO LINE=NO.
00131      IF COBX=40 NOT * SPACES MOVE 'C' TO RPTWS=1.
00132      MC15.
00133      MOVE SPACE TO COBX=80.
00134      MOVE RPTWSX80 TO RPTR80.
00135      MOVE RPTWSX53 TO RPTR53.
00136      WRITE RPTH AFTER POSITIONING 1 LINES.
00137      GO TO MC10.
00138      MC20.
00139      CLOSE COB AVX RPT.
00140      GOBACK.

```

CONDENSED LISTING

69	OPEN	0007DC	70	OPEN	000826	71	WRITE	00086A
72	MOVE	00088E	73	MOVE	0008C2	75	READ	0008C6
75	GO	0008EC	76	READ	0008F2	76	GO	000912
77	IF	000918	78	MOVE	00092C	79	MOVE	000936
80	MOVE	00093C	82	MOVE	000952	83	MOVE	000958
84	IF	00095C	84	MOVE	000966	85	IF	00096A
85	MOVE	000974	86	IF	000978	87	MOVE	000982
88	MOVE	000986	89	ADD	00098A	90	MOVE	0009A0
91	WRITE	00098B	92	WRITE	000A0C	93	MOVE	000A60
94	GO	000A66	95	IF	000A6C	96	MOVE	000A76
97	MOVE	000A7A	98	ADD	000A7E	99	MOVE	000A94
100	WRITE	000AAC	101	WRITE	000B00	102	WRITE	000B54
103	MOVE	000BA8	104	MOVE	000BAE	105	GO	000B84
106	IF	000B8A	107	ADD	000BCC	108	MOVE	000BE2
109	WRITE	000BFA	110	WRITE	000C4E	111	WRITE	000CA2
112	WRITE	000CF6	113	MOVE	000D4A	114	ADD	000D50
115	IF	000D66	116	MOVE	000D70	117	MOVE	000D74
119	ADD	000D84	120	MOVE	000D9A	121	IF	000DA0
122	MOVE	000DAA	123	WRITE	000DB4	124	GO	000DFE
125	IF	000E04	125	GO	000E0E	126	IF	000E14
127	MOVE	000E28	128	MOVE	000E38	130	MOVE	000E48
131	IF	000E52	131	MOVE	000E5C	133	MOVE	000E60
134	MOVE	000E64	135	MOVE	000E6A	136	WRITE	000E70
137	GO	000EB4	139	CLOSE	000EBA	140	GOBACK	000FBA

PRODUCED BY SMS/360 PRE-A(2.3)/BETA
 A PROPRIETARY PROGRAM OF
 BOOLE AND BARBAGE, INC.
 18990 HOMESTEAD ROAD
 CUPERTINO, CALIFORNIA 95014
 408-255-1200

CODE ACTIVITY REPORT

PAGE 1

ANALYST'S NAME=
 LOCATION=

REPORT DATE 17 JAN 75
 REPORT TIME 15.24.08
 ANALYSIS RUN NUMBER=

EXTRACTOR DATA SETS

DATA SET NAME FT04F001
 JOB NAME AVSXX
 STEP NAME PPEXTRCT
 CONTROL MODULE NAME MAIN
 EXTRACTION
 DATE 17 JAN 75
 START TIME 15.23.33
 END TIME 15.23.39
 REGION ROUNDS(ABSOLUTE)
 FWA 295800
 LWA 281000
 SAMPLE ROUNDS
 FWA 000000
 LWA 018800
 SAMPLE INTERVAL (MS) 16
 SAMPLE COUNTS
 BELOW FWA 0
 IN ROUNDS 60
 ABOVE LWA 94
 DSOX WAIT 157
 SAMPLE INTERRUPTS
 INCLUDING WAIT 311
 EXCLUDING WAIT 104
 TOTAL INTERRUPTS 318
 CONFIGURATION ID
 EXTRACTOR ID PPEPE2.1
 EXTRACTOR SERIAL NO. 0
 WEIGHTING FACTOR 1
 STUDY CRITERIA
 START TIME 15.23.33
 END TIME 15.23.39

* * * * *

DISTRIBUTION OF ACTIVITY (EXCLUDING DSOX WAIT)

	ACTIVE	WAIT	TOTAL
BELOW LOWER SAMPLE BOUNDARY	0.0	0.0	0.0
WITHIN SAMPLE BOUNDARIES	9.00	10.29	19.29
ABOVE UPPER SAMPLE BOUNDARY	24.44	5.79	30.23
TOTAL	33.44	16.08	49.52

* * * * *

DISTRIBUTION OF DSNW WAIT

DATA SET NAME	PERCENT OF ACTIVITY
SYSAFND	0.0
SYSDUT	0.0
SYSD01	6.75
SYSD02	9.97
SYSD04	33.76

TOTAL	50.48

* * * * *

MODULE MAP (EXCLUDING WAIT)

MODULE NAME	FIRST BYTE ADDRESS	LAST BYTE ADDRESS	PERCENT OF RUN TIME	MODULES WITH OVERLAYS	MODULES FOR WHICH REPORTS ARE PROVIDED
MAIN	001E58	003800	26.92		X
IGGQ19AR	164EF0	165000	0.0		
IGGQ19AQ	165058	165100	0.0		

* * * * *

STUDY REPORT FOR MODULE MAIN

A. BOUNDS FOR THE STUDY REPORT

LOWER BOUND = 000000
UPPER BOUND = 01B800

NUMBER OF BYTES PER UNIT INTERVAL = 32, 00020(HEX)

B. DISTRIBUTION OF EXECUTED INSTRUCTIONS (EXCLUDING WAIT)

	ACTIVE	WAIT	TOTAL
BELOW LOWER STUDY BOUNDARY	0.0	0.0	0.0
WITHIN STUDY BOUNDARIES	26.92	0.0	26.92
ABOVE UPPER STUDY BOUNDARY	0.0	0.0	0.0
TOTAL	26.92	0.0	26.92

C. PERCENTAGE OF TIME SPENT EXECUTING SVC'S = 24.04

D. THE MOST FREQUENTLY EXECUTED INTERVALS (EXCLUDING WAIT)

STARTING LOCATION	ENDING LOCATION	PERCENT OF RUN TIME
000F40	000F5F	10.58
000820	00083F	6.73
000840	00085F	6.73
000860	0008FF	0.96
0008A0	0008BF	0.96
001140	00115F	0.96

E. CODE EXECUTION FREQUENCY FOR EACH INTERVAL (EXCLUDING WAIT)

STARTING LOCATION	ENDING LOCATION	INTERVAL PERCENT	CUMULATIVE PERCENT	HISTOGRAM - PERCENT OF TIME (EACH * = 0.3 PERCENT)
				0 3.0 6.0 9.0 12.0 15.0
000000	00081F	0.0	0.0	"
000820	00083F	6.73	6.73	*****
000840	00085F	6.73	13.46	*****
000860	0008DF	0.0	13.46	"
0008E0	0008FF	0.96	14.42	***
000900	00089F	0.0	14.42	"
0008A0	0008BF	0.96	15.38	***

74

STARTING LOCATION	ENDING LOCATION	INTERVAL PERCENT	CUMULATIVE PERCENT	HISTOGRAM - PERCENT OF TIME (EACH * = 0.3 PERCENT)						
				0	3.0	6.0	9.0	12.0	15.0	
.0008C0	.000F3F	0.0	15.38	=						
.000F40	.000F5F	10.58	25.96	=	*****					
.000F60	.00113F	0.0	25.96	=						
.001140	.00115F	0.96	26.92	=	***					
.001160	.018800	0.0	26.92	=						

. THE INTERVAL SIZE CONSISTS OF MORE THAN ONE UNIT INTERVAL.

* * * * *

58

FORTRAN PROGRAM INPUT

LINE
NUMBER

INPUT FORTRAN STATEMENT

```

1  C..... REF: UCLA DEMO 14.0
2  C FOR DESCRIPTION OF METHOD USED IN THIS PROGRAM, SEE **THE QR
3  C TRANSFORMATION** BY FRANCIS, COMPUTER JOURNAL, VOL. 4 PP. 265,332
4  C START HERE TO PRODUCE UPPER HESSENBURG MATRIX
5      DIMENSION A(10,10),VEC(8), EIGVEC(4,4), INT(10),CC(10),SS(10)
6      DIMENSION Z(10),D(10,10),BB(10,10)
7      DATA A / 0.1,5.8,1.9,3.2,6*0.0,2.8,3.9,0.7,0.1,6*0.0,
8              Z 6.2,3.5,0.6,1.2,6*0.0,9.9,6.2,1.7,3.8 /
9      NMAX = 10
10     NN = 4
11     NVEC = 4
12     N=NN
13     K=N-2
14     DO1J=1,K
15         K1=J+1
16         K2=J+2
17         GREAT= ABS(A(K1,J))
18         MAX=K1
19         DO2I=K2,N
20             AB= ABS(A(I,J))
21             IF (GREAT.GE.AB) GC TO 2
22         3 GREAT=AB
23         MAX=I
24     2 CONTINUE
25  C INT(I) GIVES INFORMATION ABOUT INTERCHANGES
26     INT(J)=MAX
27     IF (MAX.EQ.K1) GO TO 5
28     4 DO6L=J,N
29         E=A(K1,L)
30         A(K1,L)=A(MAX,L)
31     6 A(MAX,L)=E
32     DO7L=1,N
33         E=A(L,K1)
34         A(L,K1)=A(L,MAX)
35     7 A(L,MAX)=E
36  C INTERCHANGE DONE
37     5 DO8I=K2,N
38         IF (A(I,J).EQ.0.) GC TO 8
39         A(I,J)=-A(I,J)/A(K1,J)
40         DO11M=K1,N
41     11 A(I,M)=A(I,M)+A(I,J)*A(K1,M)
42     8 CONTINUE
43  C LEFT MULT. DONE
44     DO 2000 M=K2,N
45         IF (A(M,J).EQ.0.) GC TO 2000
46     10 DO 2001 I=1,N
47     2001 A(I,K1)=A(I,K1)-A(I,M)*A(M,J)
48     2000 CONTINUE
49     1 CONTINUE
50  C RIGHT MULT. DONE
51  C HESSENBURG MATRIX NOW FORMED AS A(I,J). MULTIPLIERS STORED WITH IT
52     DO 105 I=1,N

```

 ORIGINAL PAGE IS
OF POOR QUALITY

FORTRAN PROGRAM INPUT

LINE
NUMBER

INPUT FORTRAN STATEMENT

```

53      DO 105 J=1,N
54      105 D(I,J)=A(I,J)
55      C PUT APPROPRIATE ZEROS WHERE MULTIPLIERS WERE STORED
56      DO 106 I=3,N
57      L=I-2
58      DO 106 J=1,L
59      106 A(I,J)=0.0
60      K1=N+1
61      R=0.0
62      C=0.0
63      DO33 I=1,N
64      R=R+ ABS(A(1,I))
65      33 C=C+ ABS(A(1,N))
66      DO34 I=2,N
67      RS=0.0
68      CS=0.0
69      L=I-1
70      DO35 J=L,N
71      RS=RS+ ABS(A(I,J))
72      K1J=K1-J
73      K1I=K1-I
74      35 CS=CS+ ABS(A(K1J,K1I))
75      IF (RS.LE.R) GO TO 36
76      37 R=RS
77      36 IF (CS.LE.C) GO TO 34
78      44 C=CS
79      34 CONTINUE
80      IJK=1
81      IF (R.GE.C) GO TO 54
82      C DELTA IS CRITERION FOR STOPPING AN ITERATION
83      53 DELTA=R*1.0E-11
84      C DEL IS ARBITRARY SHIFT
85      DEL=R/17.1829
86      GO TO 153
87      54 DELTA=C*1.0E-11
88      DEL=C/17.1829
89      C DELTA COMPUTED
90      C PRODUCE A-PI, ITERATE ON THIS TO BREAK UP EIGENVALUES OF EQUAL MODULUS
91      153 DO 107 I=1,N
92      107 A(I,I)=A(I,I)-DEL
93      55 IF (N.NE.1) GO TO 48
94      C SUBROUTINE CALCEV CALCULATES EIGENVALUES OF 1X1 OR 2X2 MATRICES.
95      C PI IS ADDED BACK IN CALCEV
96      47 CALL CALCEV(A,NMAX,DEL,VEC,IJK,A,I)
97      GO TO 76
98      48 IF (N.NE.2) GO TO 80
99      49 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
100     GO TO 76
101     C QR TRANSFORMATION STARTS HERE
102     C INITIALIZE FOR DETERMINING SUBSEQUENT SHIFTS
103     80 AR1=0.0
104     AR3=0.0

```

N
H

FORTRAN PROGRAM INPUT

LINE
NUMBER

INPUT FORTRAN STATEMENT

ORIGINAL PAGE IS
OF POOR QUALITY

```

105      AR4=0.0
106      C RE-ENTER AFTER REALIT OR COMPT SUPROUTINES CALLED
107      C SELECT FIRST ((ZERO)) SUB-DIAGONAL ELEMENT FROM BOTTCM
108          61 L=N
109          DO56 I=2,N
110              N2I=N+2-I
111              IF( ABS(A(N2I,N2I-1)).LT.DELTA)GO TO 57
112          56 L=L-1
113      C VALUE OF L SELECTED
114          57 IF (L.NE.N) GO TO 59
115          58 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
116              N=N-1
117              GO TO 48
118          59 IF ((L-N+1).NE.0) GO TO 75
119          60 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
120              N=N-2
121              GO TO 55
122      C DETERMINE EIGENVALUES OF 2X2 MATRIX -- ROW 1 = A(N-1,N-1)  A(N-1,N)
123      C                                     ROW 2 = A(N,N-1)      A(N,N)
124      C AND BASE SHIFT ON THESE
125          75 B=A(N-1,N-1)+A(N,N)
126          C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
127          C1=A(N-1,N-1)-A(N,N)
128          DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)
129          IF (DISC.GE.0.) GO TO 63
130          62 R1=0.5*B
131              R2=.5* SQRT( ABS(DISC))
132              ANU1=(R1-AR3)*(R1-AR3)+(R2-AR4)*(R2-AR4)
133              DENCM=AR3*AR3+AR4*AR4
134              IF (DENCM.NE.0.) GO TO 200
135          201 RHO=0.0
136              SIGMA=0.0
137              GO TO 203
138          200 RHC=C
139              SIGMA=B
140          203 AR3=R1
141              AR4=R2
142              CALL COMPT(A,NMAX,L,N,RHC,SIGMA)
143              GO TO 61
144          63 IF (B.GE.0.) GO TO 65
145          64 R1=.5*(B- SQRT(DISC))
146              GO TO 66
147          65 R1=.5*(B+ SQRT(DISC))
148              IF (R1.NE.0.) GO TO 66
149          83 R2=0.0
150              GO TO 84
151          66 R2=C/R1
152      C ROOTS OF 2*2 MATRIX FOUND
153          84 IF( ABS(A(N,N)-R1).GE. ABS(A(N,N)-R2))GO TO 68
154          67 AR2=R1
155              GO TO 69
156          68 AR2=R2

```

W
H

FORTRAN PROGRAM INPLT

LINE
NUMBER

INPUT FORTRAN STATEMENT

```

157      69 IF (AR1.NE.0.) GO TO 71
158      72 SHIFT=0.0
159      GO TO 73
160      71 SHIFT=AR2
161      73 AR1=AR2
162      CALL REALIT(A,NMAX,L,N,SHIFT,CC,SS)
163      GO TO 61
164      C ARE VECTORS NEEDED
165      76 IF (NVEC.EQ.0) GO TO 600
166      C INVERSE ITERATION IS USED FOR EIGENVECTORS
167      550 SMALL=DELTA*1.0E-30
168      NUM=0
169      JI=1
170      NN=NN+NN
171      C C1 IS REAL PART OF EIGENVALUE
172      513 C1=VEC(JI)
173      C C2 IS COMPLEX PART OF EIGENVALUE
174      C2=VEC(JI+1)
175      NUM=NUM+1
176      C REPLACE HESSENBERG MATRIX AND MULTIPLIERS
177      DO 570 I=1,NN
178      DO 570 J=1,NN
179      570 A(I,J)=B(I,J)
180      IF (C2.NE.0.) GO TO 561
181      CALL RVEC(A,NMAX,INT,C1,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS)
182      JI=JI+2
183      IF(NVEC.EQ.NUM)GO TO 600
184      GO TO 513
185      561 CALL INVEC(A,NMAX,INT,C1,C2,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS,Z,BB)
186      C NEXT STATEMENT IGNORES CONJUGATE COMPLEX EIGENVALUES
187      JI=JI+4
188      IF(NVEC.EQ.NUM)GO TO 600
189      GO TO 513
190      600 CONTINUE
191      WRITE(6,700) (VEC(I),I=1,NN)
192      DO 20 I=1,NN
193      20 WRITE(6,700) (EIGVEC(I,J),J=1,NN)
194      700 FORMAT(1H,5E20.8 / 5E20.8)
195      END

```

AFR1044

LINE NUMBER 193

*** NO RETURN OR STOP FOLLOWING LAST STATEMENT ***

44

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTRAN STATEMENT

1	C	*****	ENTRY *****
---	---	-------	-------------

2	C	REF: UCLA CEMD 14.0
---	---	-------	---------------------

3	C	FOR DESCRIPTION OF METHOD USED IN THIS PROGRAM, SEE **THE QR	
4	C	TRANSFORMATION** BY FRANCIS, COMPUTER JOURNAL, VOL. 4 PP. 265,332	

5	C	START HERE TO PRODUCE UPPER HESSENBURG MATRIX	
---	---	---	--

6		DIMENSION A(10,10),VEC(8), EIGVEC(4,4), INT(10),CC(10),SS(10)	
---	--	---	--

7		DIMENSION Z(10),C(10,10),RR(10,10)	
---	--	------------------------------------	--

8		DATA A / 0.1,5.8,1.9,3.2,6*0.0,2.8,3.9,0.7,0.1,6*0.0, 6.2,3.5,	
---	--	--	--

9	1	0.6,1.2,6*C,0,9.9,6.2,1.7,3.8 /	
---	---	---------------------------------	--

10	C	*****	LOGIC START *****
----	---	-------	-------------------

9	11		NMAX = 10
---	----	--	-----------

10	12		NN = 4
----	----	--	--------

11	13		NVEC = 4
----	----	--	----------

12	14		N=NN
----	----	--	------

13	15		K=N-2
----	----	--	-------

14	16		DO FOR J=1,K
----	----	--	--------------

15	17	1	K1=J+1
----	----	---	--------

16	18	1	K2=J+2
----	----	---	--------

17	19	1	GREAT= ABS(A(K1,J))
----	----	---	---------------------

18	20	1	MAX=K1
----	----	---	--------

19	21	1	DO FOR I=K2,N
----	----	---	---------------

20	22	2	AB= ABS(A(I,J))
----	----	---	-----------------

21	23	2	IF (GREAT.LT.AB)
----	----	---	------------------

22	24	3	GREAT=AB
----	----	---	----------

23	25	3	MAX=I
----	----	---	-------

26	26	2	END IF
----	----	---	--------

27	27	1	END DO FOR
----	----	---	------------

28	28	1	C	INT(I) GIVES INFORMATION ABOUT INTERCHANGES
----	----	---	---	---

26	29	1	INT(J)=MAX
----	----	---	------------

27	30	1	IF (MAX.NE.K1)
----	----	---	----------------

28	31	2	DO FOR L=J,N
----	----	---	--------------

29	32	3	E=A(K1,L)
----	----	---	-----------

30	33	3	A(K1,L)=A(MAX,L)
----	----	---	------------------

31	34	3	A(MAX,L)=E
----	----	---	------------

35	35	2	END DO FOR
----	----	---	------------

32	36	2	DO FOR L=1,N
----	----	---	--------------

33	37	3	E=A(L,K1)
----	----	---	-----------

STRUCTURED SPOTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SPOTRAN STATEMENT

34	38	3	
35	39	3	
	40	2	

```

      A(L,K1)=A(L,MAX)
      A(L,MAX)=E
    END DO ECR

```

	41	1	
--	----	---	--

END IE

37	42	1	C
38	43	1	
39	44	2	
	45	3	

INTERCHANGE DONE

```

    DO FOR I=K2,N
      IE (A(I,J).NE.0.)
        A(I,J)=-A(I,J)/A(K1,J)
    END DO FOR

```

40	46	3	
41	47	4	
	48	3	

```

    DO FOR M=K1,N
      A(I,M)=A(I,M)+A(I,J)*A(K1,M)
    END DO FOR

```

	49	2	
	50	1	

```

      END IE
    END DO FOR

```

44	51	1	C
45	52	1	
	53	2	

LEFT MULT. DONE

```

    DO FOR M=K2,N
      IE (A(M,J).NE.0.)

```

46	54	3	
47	55	4	
	56	3	

```

      DO FOR I=1,N
        A(I,K1)=A(I,K1)-A(I,M)*A(M,J)
      END DO FOR

```

	57	2	
	58	1	

```

      END IE
    END DO FOR

```

	59		
--	----	--	--

END DO FOR

	60	C	
52	61	C	
	62		

```

    RIGHT MULT. DONE
    HESSENBURG MATRIX NOW FORMED AS A(I,J). MULTIPLIERS STORED WITH IT
    DO FOR I=1,N

```

53	63	1	
54	64	2	
	65	1	

```

      DO FOR J=1,N
        B(I,J)=A(I,J)
      END DO FOR

```

5
E

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTRAN STATEMENT

1

66

END DO FOR

67

C

PUT APPROPRIATE ZEROS WHERE MULTIPLIERS WERE STORED

56

68

DO FOR I=3,N

57

69

1

L=I-2

58

70

1

DO FOR J=1,L

59

71

2

A(I,J)=0.0

72

1

END DO FOR

73

END RC FOR

60

74

K1=N+1

61

75

R=0.0

62

76

C=0.0

63

77

DO FOR I=1,N

64

78

1

R=R+ ABS(A(I,1))

65

79

1

C=C+ ABS(A(I,N))

80

END DO FOR

66

81

DO FOR I=2,N

67

82

1

RS=0.0

68

83

1

CS=0.0

69

84

1

L=I-1

70

85

1

DO FOR J=L,N

71

86

2

RS=RS+ ABS(A(I,J))

72

87

2

K1J=K1-J

73

88

2

K1I=K1-I

74

89

2

CS=CS+ ABS(A(K1J,K1I))

90

1

END DO FOR

75

91

1

IE (RS.GT.R)

76

92

2

R=RS

93

1

END IE

77

94

1

IE (CS.GT.C)

78

95

2

C=CS

H7

STRUCTURED SFORTRAN PROGRAM

INPUT
LINE

OUTPUT
LINE

NEST
LEVEL

OUTPUT SFORTRAN STATEMENT

INPUT LINE	OUTPUT LINE	NEST LEVEL	STATEMENT
	96	1	END IE
	97		END DO FOR
80	98		IJK=1
81	99		IE (R.LT.C)
	100	1	C DELTA IS CRITERION FOR STOPPING AN ITERATION
83	101	1	DELTA=R*1.0E-11
	102	1	C DEL IS ARBITRARY SHIFT
85	103	1	DEL=R/17.1829
	104		ELSE
87	105	1	DELTA=C*1.0E-11
88	106	1	DEL=C/17.1829
	107		END IE
	108		C DELTA COMPUTED
	109		C PRODUCE A-PI, ITERATE ON THIS TO BREAK UP EIGENVALUES OF EQUAL MODULUS
91	110		DO FOR I=1,N
92	111	1	A(I,I)=A(I,I)-DEL
	112		END DO FOR
	113	113	DO FOREVER
93	114	1	IE (N.EQ.1)
	115	2	C SUBROUTINE CALCEV CALCULATES EIGENVALUES OF 1X1 OR 2X2 MATRICES.
	116	2	C PI IS ADDED BACK IN CALCEV
96	117	2	CALL CALCEV(A,NMAX,DEL,VEC,IJK,I,1)
	118	2	UNDO 113
	119	1	ELSE
	120	2	120 DO FOREVER
98	121	3	IE (N.EQ.2)
99	122	4	CALL CALCEV(A,NMAX,DEL,VEC,IJK,I,2)
	123	4	UNDO 113
	124	3	END IE
	125	3	C OR TRANSFORMATION STARTS HERE
	126	3	C INITIALIZE FOR DETERMINING SUBSEQUENT SHIFTS
103	127	3	AP1=0.0
104	128	3	AP3=0.0
105	129	3	AR4=0.0
	130	3	DO FOREVER
108	131	4	C RE-ENTER AFTER REALIT OR COMPT SUBROUTINES CALLED
	132	4	C SELECT FIRST ((ZERO)) SUB-DIAGONAL ELEMENT FROM BOTTOM
	133	4	L=N

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTRAN STATEMENT

109	134	4		DO FOR I=2,N
110	135	5		N2I=N+2-1
111	136	5		UNDO IF (ABS(A(N2I,N2I-1)).LT.DELTA)
112	137	5		L=L-1
	138	4		END DO FOR
114	139	4	C	UNDO IF (L.EQ.N) VALUE OF L SELECTED
118	141	4		UNDO 120 IF ((L-N+1).EQ.0)
	142	4	C	DETERMINE EIGENVALUES OF 2X2 MATRIX -- ROW 1 = A(N-1,N-1) A(N-1,N)
	143	4	C	ROW 2 = A(N,N-1) A(N,N)
	144	4	C	AND BASE SHIFT ON THESE
125	145	4		B=A(N-1,N-1)+A(N,N)
126	146	4		C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
127	147	4		C1=A(N-1,N-1)-A(N,N)
128	148	4		DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)
129	149	4		IF (DISC.GE.0.)
144	150	5		IF (B.GE.0.)
147	151	6		R1=.5*(B+SQRT(DISC))
148	152	6		IF (R1.EQ.0.)
149	153	6		R2=0.0
	154	6		ELSE
151	155	7		R2=C/R1
	156	6		END IF
	157	5		ELSE
145	158	6		R1=.5*(B-SQRT(DISC))
151	159	6		R2=C/R1
	160	5		END IF
	161	5	C	ROOTS OF 2*2 MATRIX FOUND
153	162	5		IF (ABS(A(N,N)-R1).LT. ABS(A(N,N)-R2))
154	163	6		AR2=R1
	164	5		ELSE
156	165	6		AR2=R2
	166	5		END IF
157	167	5		IF (AR1.EQ.0.)
158	168	6		SHIFT=0.0
	169	5		ELSE
160	170	6		SHIFT=AR2
	171	5		END IF
161	172	5		AR1=AR2
162	173	5		CALL REALIT(A,NMAX,L,N,SHIFT,CC,SS)
	174	4		ELSE
130	175	5		R1=0.5*B
131	176	5		R2=.5*SQRT(ABS(DISC))
132	177	5		ANUM=(R1-AR3)*(R1-AR3)+(R2-AR4)*(R2-AR4)
133	178	5		DENOM=AR3*AR3+AR4*AR4

ORIGINAL PAGE IS
OF POOR QUALITY

H6

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTRAN STATEMENT

134	179	5	IF (DENOM.EQ.0.)
135	180	6	RHO=0.0
136	181	6	SIGMA=0.0
	182	5	ELSE
138	183	6	RHO=C
139	184	6	SIGMA=B
	185	5	END IF
140	186	5	AR3=R1
141	187	5	AR4=R2
142	188	5	CALL COMPT(A,NMAX,L,N,RHO,SIGMA)
	189	4	END IF
	190	3	END DO FOREVER

115	191	3	CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
116	192	3	N=N-1
	193	2	END DO FOREVER

	194	1	END IF
119	195	1	CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
120	196	1	N=N-2
	197		END DO FOREVER

INPUT LINE	OUTPUT LINE	NEST LEVEL	STATEMENT
	198		C ARE VECTORS NEEDED
165	199		IF (NVEC.NE.0)
	200	1	C INVERSE ITERATION IS USED FOR EIGENVECTORS
167	201	1	SMALL=DELTA*1.0E-30
168	202	1	NUM=0
169	203	1	J1=1
170	204	1	AP=NN+AN

	205	1	DO FOREVER
	206	2	C C1 IS REAL PART OF EIGENVALUE
172	207	2	C1=VEC(J1)
	208	2	C C2 IS COMPLEX PART OF EIGENVALUE
174	209	2	C2=VEC(J1+1)
175	210	2	NUM=NUM+1

	211	2	C REPLACE HESSENBURG MATRIX AND MULTIPLIERS
177	212	2	DO FOR I=1,NN

178	213	3	DO FOR J=1,NN
179	214	4	A(I,J)=0(I,J)
	215	3	END DO FOR

	216	2	END DO FOR
--	-----	---	------------

101

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTRAN STATEMENT

180	217	2		IE (C2.NE.C.)
185	218	3		CALL INVEC(A,NMAX,INT,C1,C2,SPALL,NN,NUM,EIGVEC,NVEC,
	219	3	1	CC,SS,Z,BB)
	220	3	C	NEXT STATEMENT IGNORES CONJUGATE COMPLEX EIGENVALUES
187	221	3		JI=JI+4
188	222	3		UNDO IE (NVEC,FQ,NUM)
	223	2		ELSE
181	224	3		CALL RVEC(A,NHAX,INT,C1,SMALL,NN,NUM,EIGVEC,NVEC,CC,
	225	3	1	SS)
182	226	3		JI=JI+2
183	227	3		UNDO IE (NVEC,FQ,NUM)
	228	2		END IE
	229	1		END DO EDOEVER
	230			END IE
191	231			WRITE (6, 90010) (VEC(I),I=1,NN)
192	232			DO FOR I=1,NN
193	233	1		WRITE (6, 90010) (EIGVEC(I,J),J=1,NN)
	234			END DO FOR
	235			STOP

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT
LINE

NEST
LEVEL

OUTPUT SFORTTRAN STATEMENT

236

C

*******FORMAT STATEMENTS*******

194

237

```
90010 FORMAT (1H ,5E20.8 / 5E20.8)
```

238

C

```
***** END PROGRAM UNIT *****
```

239

END

124

FORMAT CROSS_REFERENCE TABLE

FORMAT STATEMENT
NUMBER

LOCATION

REFERENCES

90010

237

231

233

ORIGINAL PAGE IS
OF POOR QUALITY

FORTRAN PROGRAM INPUT

LINE NUMBER	INPUT FORTRAN STATEMENT
1	C..... REF: UCLA DEMO 14.1
2	SUBROUTINE CALCEV(A,NMAX,DEL,VEC,IJK,N,M)
3	DIMENSION A(NMAX,NMAX),VEC(1)
4	IF (M.NE.1) GO TO 2
5	1 A1=0.0
6	A(N,N)=A(N,N)+DEL
7	C STORE EIGENVALUES IN VEC
8	VEC(IJK)=A(N,N)
9	VEC(IJK+1)=A1
10	IJK=IJK+2
11	RETURN
12	2 B=A(N-1,N-1)+A(N,N)
13	C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
14	C1=A(N-1,N-1)-A(N,N)
15	DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)
16	IF (DISC.LT.0.) GO TO 9
17	4 A1=0.0
18	IF (B.GE.0.) GO TO 6
19	R1=0.5*(B- SQRT(DISC))
20	GO TO 7
21	6 R1=0.5*(B+ SQRT(DISC))
22	IF (R1.NE.0.) GO TO 7
23	10 R2=0.0
24	GO TO 11
25	7 R2=C/R1
26	11 R1=R1+DEL
27	R2=R2+DEL
28	VEC(IJK)=R2
29	VEC(IJK+1)=A1
30	VEC(IJK+2)=R1
31	VEC(IJK+3)=A1
32	GO TO 8
33	9 P1=0.5*B
34	A1=-.5* SQRT(ABS(DISC))
35	R1=R1+DEL
36	VEC(IJK)=R1
37	VEC(IJK+1)=-A1
38	VEC(IJK+2)=R1
39	VEC(IJK+3)=A1
40	8 IJK=IJK+4
41	RETURN
42	END

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL	OUTPUT SFORTRAN STATEMENT
	1	C	***** CALCEV ENTRY *****
	2	C REF: UCLA DEMO 14.1
2	3		SUBROUTINE CALCEV(A,NMAX,DEL,VEC,IJK,N,M)
3	4		DIMENSION A(NMAX,NMAX),VEC(1)
	5	C	***** LOGIC START *****
4	6		IE (M.EQ.1)
5	7	1	AI=0.0
6	8	1	A(N,N)=A(N,N)+DEL
	9	1	STORE EIGENVALUES IN VEC
8	10	1	VEC(IJK)=A(N,N)
9	11	1	VEC(IJK+1)=AI
10	12	1	IJK=IJK+2
11	13	1	RETURN
	14		END IE
12	15		B=A(N-1,N-1)+A(N,N)
13	16		C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
14	17		C1=A(N-1,N-1)-A(N,N)
15	18		DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)
16	19		IE (DISC.GE.0.)
17	20	1	AI=0.0
18	21	1	IE (B.GE.0.)
21	22	2	R1=0.5*(B+ SQRT(DISC))
22	23	2	IE (R1.EQ.0.)
23	24	3	R2=0.0
	25	2	ELSE
25	26	3	R2=C/R1
	27	2	END IE
	28	1	ELSE
19	29	2	R1=0.5*(B- SQRT(DISC))
25	30	2	R2=C/R1
	31	1	END IE
26	32	1	R1=R1+DEL
27	33	1	R2=R2+DEL
28	34	1	VEC(IJK)=R2
29	35	1	VEC(IJK+1)=AI
30	36	1	VEC(IJK+2)=R1
31	37	1	VEC(IJK+3)=AI
	38		ELSE
33	39	1	R1=0.5*B
34	40	1	AI=-.5* SQRT(ABS(DISC))
35	41	1	R1=R1+DEL
36	42	1	VEC(IJK)=R1
37	43	1	VEC(IJK+1)=-AI
38	44	1	VEC(IJK+2)=R1
39	45	1	VEC(IJK+3)=AI
	46		END IE
40	47		IJK=IJK+4
41	48		RETURN

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTRAN STATEMENT

49

←
C

***** END PROGRAM UNIT *****

50

END

ORIGINAL PAGE IS
OF POOR QUALITY

16H

FORTRAN PROGRAM INPUT

LINE NUMBER	INPUT FORTRAN STATEMENT
----------------	-------------------------

1	C..... REF: UCLA DEMO 14.2
2	SUBROUTINE REALIT(A,NMAX,L,N,SHIFT,C,S)
3	DIMENSION A(NMAX,NMAX),C(1),S(1)
4	IF (SHIFT.EQ.0.) GO TO 2
5	1 DO3 I=L,N
6	3 A(I,I)=A(I,I)-SHIFT
7	2 M=N-1
8	DO4 I=L,M
9	DENOM= SQRT(A(I,I)**2+A(I+1,I)**2)
10	IF (DENOM.GT.0.) GO TO 10
11	11 C(I)=1.0
12	S(I)=0.0
13	GO TO 4
14	10 C(I)=A(I,I)/DENOM
15	S(I)=A(I+1,I)/DENOM
16	A(I,I)=DENOM
17	12 I=I+1
18	DO 35 J=I+1,N
19	F=A(I,J)
20	A(I,J)=A(I+1,J)*S(I)+F*C(I)
21	A(I+1,J)=A(I+1,J)*C(I)-F*S(I)
22	35 CONTINUE
23	4 CCNTINUE
24	C TRIANGULAR MATRIX FORMED
25	DO5 J=L,M
26	DO6 I=L,J
27	F=A(I,J)
28	A(I,J)=A(I,J+1)*S(J)+F*C(J)
29	6 A(I,J+1)=A(I,J+1)*C(J)-F*S(J)
30	A(J+1,J)=A(J+1,J+1)*S(J)
31	5 A(J+1,J+1)=A(J+1,J+1)*C(J)
32	C TRANSFORMATION COMPLETED
33	IF (SHIFT.EQ.0.) RETURN
34	7 DO9 I=L,N
35	9 A(I,I)=A(I,I)+SHIFT
36	8 RETURN
37	END

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTRAN STATEMENT

	1	C	***** REALIT ENTRY *****
--	---	---	--------------------------

	2	C REF: UCLA DEMO 14.2
--	---	---	---------------------------

2	3		SUBROUTINE REALIT(A,NMAX,L,N,SHIFT,C,S)
3	4		DIMENSION A(NMAX,NMAX),C(1),S(1)

	5	C	***** LOGIC START *****
--	---	---	-------------------------

4	6		IF (SHIFT.NE.0.)
---	---	--	------------------

5	7	1	DO FOR I=L,N
6	8	2	A(I,I)=A(I,I)-SHIFT
	9	1	END DO FOR

	10		END IF
7	11		M=N-1

8	12		DO FOR I=L,M
9	13	1	DENOM= SQRT(A(I,I)**2+A(I+1,I)**2)
10	14	1	IF (DENOM.GT.0.)
14	15	2	C(I)=A(I,I)/DENOM
15	16	2	S(I)=A(I+1,I)/DENOM
16	17	2	A(I,I)=DENOM
17	18	2	I1=I+1

18	19	2	DO FOR J=I1,N
19	20	3	F=A(I,J)
20	21	3	A(I,J)=A(I+1,J)*S(I)+F*C(I)
21	22	3	A(I+1,J)=A(I+1,J)*C(I)-F*S(I)
	23	2	END DO FOR

	24	1	ELSE
11	25	2	C(1)=1.C
12	26	2	S(1)=0.0
	27	1	END IF
	28		END DO FOR

25	29	C	DO FOR J=L,M
	30		

TRIANGULAR MATRIX FORMED

26	31	1	DO FOR I=L,J
27	32	2	F=A(I,J)
28	33	2	A(I,J)=A(I,J+1)*S(J)+F*C(J)

STRUCTURED SFORTAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL	OUTPUT SFORTAN STATEMENT
29	34	2	A(I,J+1)=A(I,J+1)*((J)-F*S(J))
	35	1	END DO FOR
30	36	1	A(J+1,J)=A(J+1,J+1)*S(J)
31	37	1	A(J+1,J+1)=A(J+1,J+1)*C(J)
	38		END DO FOR
33	39		TRANSFORMATION COMPLETED
33	40		IE (SHIFT.EQ.0.)
	41	1	RETURN
	42		END IE
34	43		DO FOR I=L,N
35	44	1	A(I,I)=A(I,I)+SHIFT
	45		END DO FOR
36	46		RETURN
	47		***** END PROGRAM UNIT *****
	48		END

ORIGINAL PAGE IS
OF POOR QUALITY

FORTRAN PROGRAM INPUT

LINE
NUMBER

INPUT FORTRAN STATEMENT

```

1      C..... REF: UCLA DEMO 14.3
2      SUBROUTINE CCMPIT(A,NMAX,L,N,RHO,SIGMA)
3      DIMENSION A(NMAX,NMAX)
4      C HOUSEHOLDER TYPE MATRICES USED INSTEAD OF ROTATIONS(SEE FRANCIS PAPER)
5      M=N-1
6      DO 1 I=L,M
7      IF (I.NE.L) GO TO 3
8      2 B1=A(I,L)*(A(I,L)-SIGMA)+A(L,L+1)*A(I+1,L)+RHO
9      B2=A(L+1,L)*(A(I,L)+A(L+1,L+1)-SIGMA)
10     B3=A(L+1,L)*A(L+2,L+1)
11     A(L+2,L)=0.0
12     GO TO 6
13     3 B1=A(I,I-1)
14     B2=A(I+1,I-1)
15     IF (I.EQ.M) GO TO 5
16     4 B3=A(I+2,I-1)
17     GO TO 6
18     5 B3=0.0
19     6 D= SQRT(B1*B1+B2*B2+B3*B3)
20     IF (B1.GE.0.) GO TO 8
21     7 D=-D
22     8 IF (D.EQ.0.) GO TO 1
23     9 C1=B2/(D+B1)
24     C2=B3/(D+B1)
25     E=2.0/(1.0+C1*C1+C2*C2)
26     DO 10 K=I,N
27     IF (I.EQ.M) GO TO 12
28     11 G=E*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))
29     A(I,K)=A(I,K)-G
30     A(I+1,K)=A(I+1,K)-C1*G
31     A(I+2,K)=A(I+2,K)-C2*G
32     GO TO 10
33     12 G=E*(A(I,K)+C1*A(I+1,K))
34     A(I,K)=A(I,K)-G
35     A(I+1,K)=A(I+1,K)-C1*G
36     10 CONTINUE
37     IF (I.EQ.L) GO TO 14
38     13 A(I,I-1)=-D
39     A(I+1,I-1)=0.0
40     IF (I.EQ.M) GO TO 14
41     15 A(I+2,I-1)=0.0
42     C ROW OPERATION COMPLETED
43     14 J=I+2
44     IF (J.LE.N) GO TO 17
45     16 J=N
46     17 DO 18 K=L,J
47     IF (I.EQ.M) GO TO 20
48     19 G=E*(A(K,I)+C1*A(K,I+1)+C2*A(K,I+2))
49     A(K,I)=A(K,I)-G
50     A(K,I+1)=A(K,I+1)-C1*G
51     A(K,I+2)=A(K,I+2)-C2*G
52     GO TO 18

```

204

FORTRAN PROGRAM INPUT

LINE
NUMBER

INPUT FORTRAN STATEMENT

```
53      20 G=E*(A(K,I)+C1*A(K,I+1))
54      A(K,I)=A(K,I)-G
55      A(K,I+1)=A(K,I+1)-C1*G
56      18 CONTINUE
57      J=I+3
58      IF (J.GT.N) GO TO 1
59      21 G=E+C2*A(I+3,I+2)
60      A(I+3,I)=-G
61      A(I+3,I+1)=-C1*G
62      A(I+3,I+2)=A(I+3,I+2)-C2*G
63      C  COLUMN OPERATION COMPLETED
64      1 CCNTINUE
65      RETURN
66      END
```

214

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL	OUTPUT SFORTRAN STATEMENT
	1		C ***** CCHPIT ENTRY *****
	2		C REF: UCLA DEMO 14.3
2	3		SUBROUTINE COMPIT(A,NMAX,L,N,RHO,SIGMA)
3	4		DIMENSION A(NMAX,NMAX)
	5		C ***** LOGIC START *****
	6		C HOUSEHOLDER TYPE MATRICES USED INSTEAD OF ROTATIONS(SEE FRANCIS PAPER)
5	7		M=N-1
6	8		DO FOR I=L,M
7	9	1	IE (I.NE.L)
13	10	2	B1=A(I,I-1)
14	11	2	B2=A(I+1,I-1)
15	12	2	IE (I.LE.M)
16	13	3	B3=A(I+2,I-1)
	14	2	ELSE
18	15	3	B3=0.0
	16	2	END IF
	17	1	ELSE
8	18	2	B1=A(I,L)*(A(L,L)-SIGMA)+A(L,L+1)*A(L+1,L)+RHO
9	19	2	B2=A(L+1,L)*(A(L,L)+A(L+1,L+1)-SIGMA)
10	20	2	B3=A(L+1,L)*A(L+2,L+1)
11	21	2	A(L+2,L)=0.0
	22	1	END IF
19	23	1	D= SQRT(B1*B1+B2*B2+B3*B3)
20	24	1	IE (B1.LT.0.)
21	25	2	C=-D
	26	1	END IF
22	27	1	IE (D.NE.0.)
23	28	2	C1=B2/(C+B1)
24	29	2	C2=B3/(C+B1)
25	30	2	E=2.0/(1.0+C1*C1+C2*C2)
26	31	2	DO FOR K=I,N
27	32	3	IE (I.NE.M)
28	33	4	G=E*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))
29	34	4	A(I,K)=A(I,K)-G
30	35	4	A(I+1,K)=A(I+1,K)-C1*G
31	36	4	A(I+2,K)=A(I+2,K)-C2*G
	37	3	ELSE
33	38	4	G=E*(A(I,K)+C1*A(I+1,K))
34	39	4	A(I,K)=A(I,K)-G
35	40	4	A(I+1,K)=A(I+1,K)-C1*G
	41	3	END IF
	42	2	END DO FOR
37	43	2	IE (I.NE.L)

STRUCTURED SFORTTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTTRAN STATEMENT

38	44	3	A(I,I-1)=-D
39	45	3	A(I+1,I-1)=0.0
40	46	3	IF (I.NE.M)
41	47	4	A(I+2,I-1)=0.0
	48	3	END IF
	49	2	END IF
	50	2	C
43	51	2	J=I+2
44	52	2	IF (J.GT.N)
45	53	3	J=N
	54	2	END IF
			ROW OPERATION COMPLETED
46	55	2	DO FOR K=L,J
47	56	3	IF (I.NE.M)
48	57	4	G=E*(A(K,I)+C1*A(K,I+1)+C2*A(K,I+2))
49	58	4	A(K,I)=A(K,I)-G
50	59	4	A(K,I+1)=A(K,I+1)-C1*G
51	60	4	A(K,I+2)=A(K,I+2)-C2*G
	61	3	ELSE
53	62	4	G=E*(A(K,I)+C1*A(K,I+1))
54	63	4	A(K,I)=A(K,I)-G
55	64	4	A(K,I+1)=A(K,I+1)-C1*G
	65	3	END IF
	66	2	END DO FOR
57	67	2	J=I+3
58	68	2	IF (J.LE.N)
59	69	3	G=E*C2*A(I+3,I+2)
60	70	3	A(I+3,I)=-G
61	71	3	A(I+3,I+1)=-C1*G
62	72	3	A(I+3,I+2)=A(I+3,I+2)-C2*G
	73	2	END IF
	74	1	END IF
	75		END DO FOR
65	76		RETURN
	77		C ***** END PROGRAM UNIT *****
	78		END

FORTRAN PROGRAM INPUT

LINE
NUMBER

INPUT FORTRAN STATEMENT

```

1      C..... REF: UCLA DEMO 14.4
2      SUBROUTINE RVEC(A,NMAX,INT,C1,SMALL,N,NUM,EIGVEC,NVEC,INTER,Y)
3      DIMENSION A(NMAX,NMAX),INTER(1),Y(1),INT(1),EIGVEC(NMAX,NMAX)
4      DO1 I=1,N
5      1 A(I,I)=A(I,I)-C1
6      DO2 I=2,N
7      IF(ABS(A(I-1,I-1)).GE. ABS(A(I,I-1)))GO TC 4
8      3 INTER(I)=I
9      J=I-1
10     DO5 K=J,N
11     B=A(I-1,K)
12     A(I-1,K)=A(I,K)
13     5 A(I,K)=B
14     GO TC 6
15     4 INTER(I)=I-1
16     6 IF (A(I,I-1).EQ.0.) GO TO 2
17     C=-A(I,I-1)/A(I-1,I-1)
18     A(I,I-1)=C
19     DO 8 K=1,N
20     8 A(I,K)=A(I,K)+C*A(I-1,K)
21     2 CCNTINUE
22     C TRIANGULAR MATRIX FORMED
23     DO9 I=1,N
24     IF(ABS(A(I,I)).GF.SMALL)GO TO 9
25     10 A(I,I)=SMALL
26     9 CONTINUE
27     C NORMALISED ((EIGENVECTOR)) OF HESSENBERG MATRIX OBTAINED FOR RHS = 1
28     DO 5555 I=1,N
29     5555 Y(I)=1.0
30     DO 8888 JJJJ=1,4
31     7777 Y(N)=Y(N)/A(N,N)
32     B=Y(N)
33     DO11 I=2,N
34     E=0.0
35     M=N+1-I
36     DO12 J=2,I,
37     L=N+J-I
38     12 E=E+Y(L)*A(M,L)
39     Y(M)=(Y(M)-E)/A(M,M)
40     IF(ABS(B).GE. ABS(Y(M)))GO TO 11
41     13 B=Y(M)
42     11 CCNTINUE
43     DO14 I=1,N
44     14 Y(I)=Y(I)/B
45     IF(JJJJ.EQ.4)GO TC 6666
46     DO15 I=2,N
47     IF (INTER(I).NE.1) GO TO 15
48     16 B=Y(I)
49     Y(I)=Y(I-1)
50     Y(I-1)=B
51     15 Y(I)=Y(I)+A(I,I-1)*Y(I-1)
52     8888 CCNTINUE

```

FORTRAN PROGRAM INPUT

LINE
NUMBER

INPUT FORTRAN STATEMENT

```
53      C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED (NOT NORMALIZED)
54      6666 M=N-1
55      DO21 I=2,M
56      J=N+1-I
57      L=J+1
58      DO20 K=L,N
59      20 Y(K)=Y(K)-Y(J)*A(K,J-1)
60      LL=INT(N-I)
61      IF (J.EQ.LL) GO TO 21
62      19 B=Y(J)
63      Y(J)=Y(LL)
64      Y(LL)=B
65      21 CONTINUE
66      C UNNORMALIZED EIGENVECTOR OF ORIGINAL MATRIX OBTAINED
67      B=0.0
68      DO22 I=1,N
69      IF( ABS(B).GE. ABS(Y(I)))GO TO 22
70      23 B=Y(I)
71      22 CONTINUE
72      DO24 I=1,N
73      24 Y(I)=Y(I)/B
74      C      NORMALIZED EIGENVECTOR OBTAINED
75      C PUT Y(I) (REAL EIGENVECTOR) IN EIGVECT(NUMOFEV,I)
76      DO 100 I=1,N
77      100 EIGVEC(NUM,I)=Y(I)
78      RETURN
79      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL	OUTPUT SFORTRAN STATEMENT
	1	C	***** RVEC ENTRY *****
	2	C REF: UCLA DEMO 14.4
2	3		SUBROUTINE RVEC(A,NMAX,INT,C1,SMALL,N,NUM,EIGVEC,NVEC,INTER,Y)
3	4		DIMENSION A(NMAX,NMAX),INTER(1),Y(1),INT(1),EIGVEC(NMAX,NMAX)
	5	C	**** LOGIC START ****
4	6	1	DO FOR I=1,N
5	7		A(I,I)=A(I,I)-C1
	8		END DO FOR
6	9	1	DO FOR I=2,N
7	10		IE (ABS(A(I-1,I-1)).LT. ABS(A(I,I-1)))
8	11		INTER(I)=1
9	12	2	J=I-1
10	13	2	DO FOR K=J,N
11	14	3	B=A(I-1,K)
12	15	3	A(I-1,K)=A(I,K)
13	16	3	A(I,K)=B
	17	2	END DO FOR
	18	1	ELSE
15	19	2	INTER(I)=I-1
	20	1	END IE
16	21	1	IE (A(I,I-1).NE.0.)
17	22	2	C=-A(I,I-1)/A(I-1,I-1)
18	23	2	A(I,I-1)=C
19	24	2	DO FOR K=I,N
20	25	3	A(I,K)=A(I,K)+C*A(I-1,K)
	26	2	END DO FOR
	27	1	END IE
	28		END DO FOR
	29	C	TRIANGULAR MATRIX FORMED
23	30		DO FOR I=1,N
24	31	1	IE (ABS(A(I,I)).LT.SMALL)

264

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTRAN STATEMENT

25

32

2

A(I,I)=SMALL

33

1

END IF

34

END DO FOR

28

35

1

C NORMALISED ((EIGENVECTOR)) OF HESSENBURG MATRIX OBTAINED FOR RHS = 1

29

36

1

DO FOR I=1,N

37

Y(I)=1.0

38

END DO FOR

30

39

1

DO FOR JJJJ=1,4

31

40

1

Y(N)=Y(N)/A(N,N)

32

41

1

B=Y(N)

33

42

1

DO FOR I=2,N

34

43

2

E=0.0

35

44

2

P=N+1-I

36

45

2

DO FOR J=2,I

37

46

3

L=N+J-I

38

47

3

E=E+Y(L)*A(M,L)

48

2

END DO FOR

39

49

2

Y(N)=(Y(N)-E)/A(M,M)

40

50

2

IF (ABS(B).LT. ABS(Y(N)))

41

51

3

B=Y(N)

52

2

END IF

53

1

END DO FOR

43

54

1

DO FOR I=1,N

44

55

2

Y(I)=Y(I)/B

56

1

END DO FOR

45

57

1

UNDO IF (JJJJ.EQ.4)

46

58

1

DO FOR I=2,N

47

59

2

IF (INTER(I).EQ.1)

48

60

3

B=Y(I)

2714

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL	OUTPUT SFORTRAN STATEMENT
49	61	3	Y(I)=Y(I-1)
50	62	3	Y(I-1)=B
	63	2	END IE
51	64	2	Y(I)=Y(I)+A(I,I-1)*Y(I-1)
	65	1	END DO FOR
	66		END DO FOR
54	67	C	EIGENVECTOR OF HESSENBURG MATRIX OBTAINED (NOT NORMALIZED)
	68		M=N-1
55	69		DO FOR I=2,M
56	70	1	J=N+1-I
57	71	1	L=J+1
	72	1	DO FOR K=L,M
58	73	2	Y(K)=Y(K)-Y(J)*A(K,J-1)
59	74	1	END DO FOR
60	75	1	LL=INT(A-1)
61	76	1	IE (J.NE.LL)
62	77	2	B=Y(J)
63	78	2	Y(J)=Y(LL)
64	79	2	Y(LL)=B
	80	1	END IE
	81		END DO FOR
67	82	C	UNNORMALIZED EIGENVECTOR OF ORIGINAL MATRIX OBTAINED
	83		B=0.0
68	84		DO FOR I=1,N
69	85	1	IE (ABS(B).LT. ABS(Y(I)))
70	86	2	B=Y(I)
	87	1	END IE
	88		END DO FOR
72	89		DO FOR I=1,N
73	90	1	Y(I)=Y(I)/B
	91		END DO FOR

284

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTRAN STATEMENT

	92	C	NORMALIZED EIGENVECTOR OBTAINED
	93	C	PUT Y(I) (REAL EIGENVECTOR) IN EIGVECT(NUMOFEV,I)
76	94		DO FOR I=1,N
77	95	1	EIGVEC(NUM,I)=Y(I)
	96		END DO FOR
78	97		RETURN
	98	C	***** END PROGRAM UNIT *****
	99		END

ORIGINAL PAGE IS
OF POOR QUALITY

FORTRAN PROGRAM INPUT

LINE
NUMBER

INPUT FORTRAN STATEMENT

```

1      C..... REF: UCLA DEMO 14.5
2      SUBROUTINE IMVEC(A,NMAX,INT,C1,AMU,SMALL,N,NUM,EIGVEC,NVEC,INTER,
3      IY,Z,B)
4      DIMENSION A(NMAX,NMAX),INTER(1),Y(1),Z(1),INT(1),B(NMAX,NMAX),
5      IEIGVEC(NMAX,NMAX)
6      DO50 I=1,N
7      DO50 J=1,N
8      50 B(I,J)=0.0
9      DO1 I=1,N
10     A(I,I)=A(I,I)-C1
11     1 B(I,I)=B(I,I)-AMU
12     DO2 I=2,N
13     C=A(I-1,I-1)*A(I-1,I-1)+B(I-1,I-1)*B(I-1,I-1)
14     D=A(I,I-1)*A(I,I-1)+B(I,I-1)*B(I,I-1)
15     IF (C.GE.D) GO TO 4
16     3 INTER(I)=I
17     G=C
18     C=D
19     D=G
20     J=I-1
21     DO5 K=J,N
22     G=A(I-1,K)
23     A(I-1,K)=A(I,K)
24     A(I,K)=G
25     G=B(I-1,K)
26     B(I-1,K)=B(I,K)
27     5 B(I,K)=G
28     GO TO 6
29     4 INTER(I)=I-1
30     6 IF (D.EQ.0.) GO TO 2
31     7 G=A(I,I-1)
32     A(I,I-1)=-(G*A(I-1,I-1)+B(I,I-1)*B(I-1,I-1))/C
33     B(I,I-1)=-(B(I,I-1)*A(I-1,I-1)-G*B(I-1,I-1))/C
34     DO8 K=I,N
35     A(I,K)=A(I,K)+A(I,I-1)*A(I-1,K)-B(I,I-1)*B(I-1,K)
36     8 B(I,K)=B(I,K)+B(I,I-1)*A(I-1,K)+A(I,I-1)*B(I-1,K)
37     2 CCATINUE
38     C      TRIANGULAR MATRIX FORMED
39     DO9 I=1,N
40     C=A(I,I)*A(I,I)+B(I,I)*B(I,I)
41     IF (C.LE.SMALL) GO TO 9
42     A(I,I)= SQRT(SMALL)
43     B(I,I)=0.0
44     9 CONTINUE
45     DO 1111 I=1,N
46     Y(I)=1.0
47     1111 Z(I)=0.0
48     DO 8888 JJJ=1,4
49     C=A(N,N)*A(N,N)+B(N,N)*B(N,N)
50     G=Y(N)
51     Y(N)=(G*A(N,N)+Z(N)*B(N,N))/C
52     Z(N)=(Z(N)*A(N,N)-C*B(N,N))/C

```

304

FORTRAN PROGRAM INPUT

LINE
NUMBER

INPUT FORTRAN STATEMENT

```

53      DO17 I=2,N
54      E=0.0
55      F=0.0
56      M=N+1-I
57      DO18 J=2,I
58      L=M+J-I
59      E=E+Y(L)*A(M,L)-Z(L)*B(M,L)
60      18 F=F+Y(L)*B(M,L)+Z(L)*A(M,L)
61      D=A(M,M)*A(M,M)+B(M,M)*B(M,M)
62      G=Y(M)
63      Y(M)=((G-E)*A(M,M)+(Z(M)-F)*B(M,M))/D
64      17 Z(M)=((Z(M)-F)*A(M,M)-(G-E)*B(M,M))/D
65      E=0.0
66      DO51 I=1,N
67      F=Y(I)*Y(I)+Z(I)*Z(I)
68      IF (E.GE.F) GO TO 51
69      52 E=F
70      L=I
71      51 CONTINUE
72      A1=Y(L)
73      A2=Z(L)
74      DO14 I=1,N
75      G=Y(I)
76      Y(I)=(G*A1+Z(I)*A2)/E
77      14 Z(I)=(Z(I)*A1-G*A2)/E
78      IF(JJJ.EQ.4) GO TO 6666
79      DO15 I=2,N
80      IF (INTER(I).NE.I) GO TO 30
81      16 G=Y(I)
82      Y(I)=Y(I-1)
83      Y(I-1)=G
84      G=Z(I)
85      Z(I)=Z(I-1)
86      Z(I-1)=G
87      30 Y(I)=Y(I)+A(I,I-1)*Y(I-1)-B(I,I-1)*Z(I-1)
88      15 Z(I)=Z(I)+B(I,I-1)*Y(I-1)+A(I,I-1)*Z(I-1)
89      8888 CONTINUE
90      C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED
91      6666 M=N-I
92      DO21 I=2,M
93      J=N+1-I
94      L=J+1
95      DO20 K=L,N
96      Y(K)=Y(K)-Y(J)*A(K,J-1)+Z(J)*B(K,J-1)
97      20 Z(K)=Z(K)-Z(J)*A(K,J-1)-Y(J)*B(K,J-1)
98      LL=INT(N-I)
99      IF (J.EQ.LL) GO TO 21
100     19 G=Y(J)
101     Y(J)=Y(LL)
102     Y(LL)=G
103     G=Z(J)
104     Z(J)=Z(LL)

```

H12

FORTRAN PROGRAM INPUT

LINE NUMBER	INPUT FORTRAN STATEMENT
105	Z(LL)=G
106	21 CONTINUE
107	C UNNORMALIZED EIGENVECTOR OBTAINED
108	E=0.0
109	DO22 I=1,N
110	F=Y(I)*Y(I)+Z(I)*Z(I)
111	IF (E.GE.F) GO TO 22
112	23 E=F
113	L=I
114	22 CONTINUE
115	A1=Y(L)
116	A2=Z(L)
117	DO24 I=1,N
118	G=Y(I)
119	Y(I)=(G*A1+Z(I)*A2)/E
120	24 Z(I)=(Z(I)*A1-G*A2)/E
121	C NORMALIZED EIGENVECTOR OBTAINED
122	C PUT Y(I) (REAL PART) IN EIGVECT(NUMOFEV,I)
123	DO 100 I=1,N
124	100 EIGVEC(NUM,I)=Y(I)
125	NUM=NUM+1
126	C PUT Z(I) (COMPLEX PART) IN EIGVECT(NUMOFEV,I)
127	DO 101 I=1,N
128	101 EIGVEC(NUM,I)=Z(I)
129	RETURN
130	END

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL	OUTPUT SFORTRAN STATEMENT
	1	C	***** IMVEC ENTRY *****
	2	C REF: UCLA DEMO 14.5
2	3		SUBROUTINE IMVEC(A,NMAX,INT,C1,AMU,SMALL,N,NUM,EIGVEC,NVEC,INTER,
	4	1	Y,Z,B)
4	5		DIMENSION A(NMAX,NMAX),INTER(1),Y(1),Z(1),INT(1),B(NMAX,NMAX),
	6	1	EIGVEC(NMAX,NMAX)
	7	C	***** LOGIC START *****
6	8		DO FOR I=1,N
7	9	1	DO FOR J=1,N
8	10	2	B(I,J)=C.0
	11	1	END DO FOR
	12		END DO FOR
9	13		DO FOR I=1,N
10	14	1	A(I,I)=A(I,I)-C1
11	15	1	B(I,I)=B(I,I)-AMU
	16		END DO FOR
12	17		DO FOR I=2,N
13	18	1	C=A(I-1,I-1)*A(I-1,I-1)+B(I-1,I-1)*B(I-1,I-1)
14	19	1	D=A(I,I-1)*A(I,I-1)+B(I,I-1)*B(I,I-1)
15	20	1	IF (C.LT.D)
16	21	2	INTER(I)=I
17	22	2	G=C
18	23	2	C=D
19	24	2	C=G
20	25	2	J=I-1
21	26	2	DO FOR K=J,N
22	27	3	G=A(I-1,K)
23	28	3	A(I-1,K)=A(I,K)
24	29	3	A(I,K)=G
25	30	3	G=B(I-1,K)
26	31	3	B(I-1,K)=B(I,K)
27	32	3	B(I,K)=G
	33	2	END DO FOR

334

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL	OUTPUT SFORTRAN STATEMENT
			ELSE
29	34	1	INTER(I)=I-1
	35	2	END IE
	36	1	IE (D.NE.0.)
30	37	1	G=A(I,I-1)
31	38	2	A(I,I-1)=-(G*A(I-1,I-1)+B(I,I-1)*B(I-1,I-1))/C
32	39	2	B(I,I-1)=-(B(I,I-1)*A(I-1,I-1)-G*B(I-1,I-1))/C
33	40	2	
			DO FOR K=I,N
34	41	2	A(I,K)=A(I,K)+A(I,I-1)*A(I-1,K)-B(I,I-1)*B(I-1,K)
35	42	3	B(I,K)=B(I,K)+B(I,I-1)*A(I-1,K)+A(I,I-1)*B(I-1,K)
36	43	3	
	44	2	END DO FOR
			END IE
	45	1	END DO FOR
	46		
			TRIANGULAR MATRIX FORMED
	47		C
39	48		DO FOR I=1,N
40	49	1	C=A(I,I)*A(I,I)+B(I,I)*B(I,I)
41	50	1	IE (C.LT.SPALL)
42	51	2	A(I,I)= SQRT(SPALL)
43	52	2	B(I,I)=C.0
	53	1	END IE
	54		END DO FOR
45	55		DO FOR I=1,N
46	56	1	Y(I)=1.0
47	57	1	Z(I)=0.0
	58		END DO FOR
48	59		DO FOR JJJ=1,4
49	60	1	C=A(N,N)*A(N,N)+B(N,N)*B(N,N)
50	61	1	G=Y(N)
51	62	1	Y(N)=(G*A(N,N)+Z(N)*B(N,N))/C
52	63	1	Z(N)=(Z(N)*A(N,N)-G*B(N,N))/C
53	64	1	DO FOR I=2,N
54	65	2	E=0.0

34H

STRUCTURED SFORTRAN PROGRAM

INPUT
LINE

OUTPUT
LINE

NEXT
LEVEL

OUTPUT SFORTRAN STATEMENT

55 66 2
56 67 2

F=0.0
N=N+1-I

57 68 2
58 69 3
59 70 3
60 71 3
61 72 2

DO FOR J=2,I
L=N+J-1
E=E+Y(L)*A(M,L)-Z(L)*B(M,L)
F=F+Y(L)*B(M,L)+Z(L)*A(M,L)
END DO FOR

61 73 2
62 74 2
63 75 2
64 76 2
65 77 1

C=A(M,N)*A(M,M)+B(M,N)*B(M,M)
G=Y(M)
Y(M)=(G-E)*A(M,M)+(Z(M)-F)*B(M,M)/D
Z(M)=(Z(M)-F)*A(M,M)-(G-E)*B(M,M)/C
END DO FOR

65 78 1

E=0.0

66 79 1
67 80 2
68 81 2
69 82 3
70 83 3
71 84 2
72 85 1

DO FOR I=1,N
F=Y(I)*Y(I)+Z(I)*Z(I)
IE (E,LT,F)
E=F
L=I
END IF
END DO FOR

72 86 1
73 87 1

A1=Y(L)
A2=Z(L)

74 88 1
75 89 2
76 90 2
77 91 2
78 92 1

DO FOR I=1,N
G=Y(I)
Y(I)=(G*A1+Z(I)*A2)/E
Z(I)=(Z(I)*A1-G*A2)/E
END DO FOR

78 93 1

UNDO IE (JJJ.EQ.4)

79 94 1
80 95 2
81 96 3
82 97 3
83 98 3
84 99 3
85 100 3

DO FOR I=2,N
IE (INTER(I).EQ.1)
G=Y(I)
Y(I)=Y(I-1)
Y(I-1)=G
G=Z(I)
Z(I)=Z(I-1)

ORIGINAL PAGE IS
OF POOR QUALITY

STRUCTURED SFORTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL	OUTPUT SFORTRAN STATEMENT
86	101	3	Z(I-1)=G
	102	2	END IF
87	103	2	Y(I)=Y(I)+A(I,I-1)*Y(I-1)-B(I,I-1)*Z(I-1)
88	104	2	Z(I)=Z(I)+B(I,I-1)*Y(I-1)+A(I,I-1)*Z(I-1)
	105	1	END DO FOR
	106		END DO FOR
91	107	C	EIGENVECTOR OF HESSENBURG MATRIX OBTAINED
	108		M=N-1
92	109		DO FOR I=2,M
93	110	1	J=N+1-I
94	111	1	L=J+1
95	112	1	DO FOR K=L,M
96	113	2	Y(K)=Y(K)-Y(J)*A(K,J-1)+Z(J)*B(K,J-1)
97	114	2	Z(K)=Z(K)-Z(J)*A(K,J-1)-Y(J)*B(K,J-1)
	115	1	END DO FOR
98	116	1	LL=INT(N-.1)
99	117	1	IF (J=NE,LL)
100	118	2	G=Y(J)
101	119	2	Y(J)=Y(LL)
102	120	2	Y(LL)=G
103	121	2	G=Z(J)
104	122	2	Z(J)=Z(LL)
105	123	2	Z(LL)=G
	124	1	END IF
	125		END DO FOR
108	126	C	UNNORMALIZED EIGENVECTOR OBTAINED
	127		E=0.0
109	128		DO FOR I=1,N
110	129	1	F=Y(I)*Y(I)+Z(I)*Z(I)
111	130	1	IF (E.LT.F)
112	131	2	E=F
113	132	2	L=I
	133	1	END IF
	134		END DO FOR
115	135		A1=Y(L)
116	136		A2=Z(L)

364

STRUCTURED SFORTTRAN PROGRAM

INPUT LINE	OUTPUT LINE	NEST LEVEL
---------------	----------------	---------------

OUTPUT SFORTTRAN STATEMENT

117	137	
118	138	1
119	139	1
120	140	1
	141	

```
DO FOR I=1,N
  G=Y(I)
  Y(I)=(G*A1+Z(I)*A2)/E
  Z(I)=(Z(I)*A1-G*A2)/E
END DO FOR
```

	142	
	143	C
123	144	
124	145	1
	146	

```
          NORMALIZED EIGENVECTOR OBTAINED
          PUT Y(I) (REAL PART) IN EIGVECT(NUMOFEV,I)
DO FOR I=1,N
  EIGVECT(NUM,I)=Y(I)
END DO FOR
```

125	147	
-----	-----	--

NUM=NUM+1

	148	
127	149	C
128	150	1
	151	

```
          PUT Z(I) (COMPLEX PART) IN EIGVECT(NUMOFEV,I)
DO FOR I=1,N
  EIGVECT(NUM,I)=Z(I)
END DO FOR
```

129	152	
-----	-----	--

RETURN

	153	
--	-----	--

```
<-----
C ***** END PROGRAM UNIT *****
```

	154	
--	-----	--

END


```

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT) #0
DOUBLE PRECISION DELTAX, DELTA, A, PROD, SUM #1
DIMENSION X(1), Y(1) #2
DIMENSION DELTA(10), A(10) #3

```

```

SEARCH FOR APPROPRIATE VALUE OF X(I) #4

```

```

11 DO 19 I=1, NPTS #5
   IF (XIN - X(I)) 13, 17, 19 #6
13 I1 = I - NTERMS/2 #7
   IF (I1) 15, 15, 21 #8
15 I1 = 1 #9
   GO TO 21 #10
17 YOUT = Y(I) #11
18 GO TO 61 #12
19 CONTINUE #13
   I1 = NPTS - NTERMS + 1 #14
21 I2 = I1 + NTERMS - 1 #15
   IF (NPTS - I2) 23, 31, 31 #16
23 I2 = NPTS #17
   I1 = I2 - NTERMS + 1 #18
25 IF (I1) 26, 26, 31 #19
26 I1 = 1 #20
27 NTERMS = I2 - I1 + 1 #21

```

```

EVALUATE DEVIATIONS DELTA #22

```

```

31 DENOM = X(I1+1) - X(I1) #23
   DELTAX = (XIN - X(I1)) / DENOM #24
   DO 35 I=1, NTERMS #25
   IX = I1 + I - 1 #26
   DELTA(I) = (X(IX) - X(I1)) / DENOM #27
35 CONTINUE #28

```

```

ACCUMULATE COEFFICIENTS A #29

```

```

40 A(1) = Y(I1) #30
41 DO 50 K=2, NTERMS #31
   PROD = 1. #32
   SUM = 0. #33
   IMAX = K - 1 #34
   IXMAX = I1 + IMAX #35
   DO 49 I=1, IMAX #36
   J = K - I #37
   PROD = PROD * (DELTA(K) - DELTA(J)) #38
49 SUM = SUM + A(J)/PROD #39
   A(K) = SUM + Y(IXMAX)/PROD #40
50 CONTINUE #41

```

```

ACCUMULATE SUM OF EXPANSION #42

```

```

51 SUM = A(1) #43
   DO 57 J=2, NTERMS #44
   PROD = 1. #45
   IMAX = J - 1 #46
   DO 56 I=1, IMAX #47
56 PROD = PROD * (DELTAX - DELTA(I)) #48
57 SUM = SUM + A(J)*PROD #49
60 YOUT = SUM #50
61 RETURN #51
END #52

```

ORIGINAL PAGE IS
OF POOR QUALITY

TESTCASE Output File ((1 15) (INTERP . OUT))

--date--

Program file: ((1 15) (INTERP . F4))
Analysis commands file: ((1 15) (INTERP . ACF))

ANALYSIS OF INTERP

Global Commands:

MAXPATHS 300;
MAXLENGTH 300;
DEFAULT LOOP ANY CONSISTENT(1-10);
DEFAULT SELECT ANY CONSISTENT;
PATH; PREDICATES;

Case 1 =====

ANALYSIS OF SECTION OF CODE THAT DETERMINES X1.
CHECK FOR NPTS=1 CASE.

Case Commands:

OUTPUT I1, NTERMS;
SELECT ALL;
7 ASSIGN NPTS=1;
23 HALT;

Case 1.1 -----

PATH: 2-12 13-24

PREDICATES:

:1 0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8	XIN = X(1) .LT. 0
:9 10	1 = (NTERMS/2) .LE. 0
:13 19	1 = ((1 + NTERMS) - 1) .LT. 0
:16 22	1 + 1 = NTERMS .LE. 0

OUTPUT:

Final Values

:13 28	I1 = 1
	NTERMS = 1

Case 1.2 -----

PATH: 8-12 18-22

PREDICATES:

```

:1 3      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .LE. 0
:13 15    1 = ((1 + NTERMS) - 1) .LT. 0
:16 22    1 + 1 = NTERMS .GT. 0

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:16 28    I1 = 1 + 1 = NTERMS
          NTERMS = NTERMS

```

Case 1.3 -----

PATH: 0-12 18-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .LE. 0
:13 19    1 = ((1 + NTERMS) - 1) .GE. 0

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:13 28    I1 = 1
          NTERMS = NTERMS

```

Case 1.4 -----

PATH: 0-10 18-24

PREDICATES:

```

:1 2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .GT. 0
:11 19    1 = ((1 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:14 22    1 + 1 = NTERMS .LE. 0

```

OUTPUT:

Final Values

```

:16 28    I1 = 1
          NTERMS = 1

```

Case 1.5 -----

PATH: 0-10 12-22

PREDICATES:

:1	0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7	0	XIN = X(1) .LT. 0
:9	10	1 - (NTERMS/2) .GT. 0
:11	19	1 - ((1 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:14	22	1 + 1 - NTERMS .GT. 0

OUTPUT:

:1	0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
----	---	--

Final Values

:14	28	I1 = 1 + 1 - NTERMS
		NTERMS = NTERMS

Case 1.6 -----

PATH: 3-10 18-19

PREDICATES:

:1	0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7	0	XIN = X(1) .LT. 0
:9	10	1 - (NTERMS/2) .GT. 0
:11	19	1 - ((1 - (NTERMS/2) + NTERMS) - 1) .GE. 0

OUTPUT:

:1	0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
----	---	--

Final Values

:11	28	I1 = 1 - (NTERMS/2)
		NTERMS = NTERMS

Case 1.7 -----

PATH: 0-8 13-14 59

PREDICATES:

:1	0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7	8	XIN = X(1) .EQ. 0

OUTPUT:

:1	0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
----	---	--

59 NOTE: Reference to uninitialized variable I1

Final Values

:10	59	I1 = I1
		NTERMS = NTERMS

Case 1.8 -----

PATH: 0-8 15 7 16-24

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:13 19    1 - ((1 + 1 - NTERMS + NTERMS) - 1) .LT. 0
:16 22    1 + 1 - NTERMS .LE. 0
```

OUTPUT:

Final Values

```
:13 23    I1 = 1
           NTERMS = 1
```

Case 1.9 -----

PATH: 2-8 15 7 16-22

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:13 19    1 - ((1 + 1 - NTERMS + NTERMS) - 1) .LT. 0
:16 22    1 + 1 - NTERMS .GT. 0
```

OUTPUT:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
```

Final Values

```
:16 28    I1 = 1 + 1 - NTERMS
           NTERMS = NTERMS
```

Case 1.10 -----

PATH: 2-8 15 7 16-19

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:13 19    1 - ((1 + 1 - NTERMS + NTERMS) - 1) .GE. 0
```

OUTPUT:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
```

Final Values

```
:13 28    I1 = 1 + 1 - NTERMS
           NTERMS = NTERMS
```

ORIGINAL PAGE IS
OF POOR QUALITY

Case 2 -----

AS IN CASE 1, CHECK FOR NPTS=2

Case Commands:

OUTPUT 11, NTERMS;
SELECT ALL;
CONSISTENCY;
7 ASSIGN NPTS-2;
23 HALT;

Case 2.1 -----

PATH: C-12 18-24

PREDICATES:

```
:1  2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7  8      XIN - X(1) .LT. 0
:9  10     1 - (NTERMS/2) .LE. 0
:13 19     2 - ((1 + NTERMS) - 1) .LT. 0
:16 22     1 + 2 - NTERMS .LE. 0
:18 28     CONSISTENCY: OK
```

OUTPUT:

Final Values

```
:18 28      I1 = 1
            NTERMS = 2
```

Case 2.2 -----

PATH: B-12 18-22

PREDICATES:

```
:1  2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7  8      XIN - X(1) .LT. 0
:9  10     1 - (NTERMS/2) .LE. 0
:13 19     2 - ((1 + NTERMS) - 1) .LT. 0
:16 22     1 + 2 - NTERMS .GT. 0
:16 28     CONSISTENCY: OK
```

OUTPUT:

```
:1  0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
```

Final Values

```
:16 28      I1 = 1 + 2 - NTERMS
            NTERMS = NTERMS
```

Case 2.3 -----

PATH: C-12 18-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .LE. 0
:13 19    2 = ((1 + NTERMS) - 1) .GE. 0
:13 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:13 28    I1 = 1
          NTERMS = NTERMS

```

Case 2.4 -----

PATH: 3-10 18-24

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .GT. 0
:11 19    2 = ((1 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:14 22    1 + 2 = NTERMS .LE. 0
:16 28    CONSISTENCY: OK

```

OUTPUT:

Final Values

```

:16 28    I1 = 1
          NTERMS = 2

```

Case 2.5 -----

PATH: 3-10 18-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .GT. 0
:11 19    2 = ((1 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:14 22    1 + 2 = NTERMS .GT. 0
:14 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:14 28    I1 = 1 + 2 = NTERMS
          NTERMS = NTERMS

```

Case 2.6 -----

PATH: 8-10 13-19

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 3      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .GT. 0
:11 19    2 = ((1 - (NTERMS/2) + NTERMS) - 1) .GE. 0
:11 28    CONSISTENCY: OK
```

OUTPUT:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
```

Final Values

```
:11 28    I1 = 1 - (NTERMS/2)
          NTERMS = NTERMS
```

Case 2.7 -----

PATH: 3-8 13-14 59

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 3      XIN = X(1) .EQ. 0
:13 59    CONSISTENCY: OK
```

OUTPUT:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
59      NOTE: Reference to uninitialized variable I1
```

Final Values

```
:10 59    I1 = I1
          NTERMS = NTERMS
```

Case 2.8 -----

PATH: 2-8 15 7-12 18-24

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 3      XIN = X(1) .GT. 0
:11 8      XIN = X(2) .LT. 0
:13 13     2 = (NTERMS/2) .LE. 0
:17 19     2 = ((1 + NTERMS) - 1) .LT. 3
:20 22     1 + 2 = NTERMS .LE. 0
:22 23     CONSISTENCY: OK
```

OUTPUT:

Final Values

```
:22 23    I1 = 1
          NTERMS = 2
```


Case 2.9 -----

PATH: 0-8 15 7-12 18-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .LT. 0
:13 18    2 - (NTERMS/2) .LE. 0
:17 19    2 - ((1 + NTERMS) - 1) .LT. 0
:23 22    1 + 2 - NTERMS .GT. 0
:28 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:28 28    I1 = 1 + 2 - NTERMS
          NTERMS = NTERMS

```

Case 2.12 -----

PATH: 3-8 15 7-12 18-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .LT. 0
:13 12    2 - (NTERMS/2) .LE. 0
:17 19    2 - ((1 + NTERMS) - 1) .GE. 0
:17 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:17 28    I1 = 1
          NTERMS = NTERMS

```

ORIGINAL PAGE IS
OF POOR QUALITY

Case 2.11 -----

PATH: 3-8 15 7-12 18-24

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .LT. 0
:13 18    2 - (NTERMS/2) .GT. 0
:15 19    2 - ((2 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:18 22    1 + 2 - NTERMS .LE. 0
:28 28    CONSISTENCY: OK

```

OUTPUT:

Final Values

:23 23

I1 = 1

NTERMS = 2

Case 2.12 -----

PATH: 3-3 15 7-10 18-22

PREDICATES:

:1 0

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

:7 8

XIN - X(1) .GT. 0

:11 8

XIN - X(2) .LT. 0

:13 10

2 - (NTERMS/2) .GT. 0

:15 19

2 - ((2 - (NTERMS/2) + NTERMS) - 1) .LT. 0

:18 22

1 + 2 - NTERMS .GT. 0

:18 22

CONSISTENCY: OK

OUTPUT:

:1 0

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:18 28

I1 = 1 + 2 - NTERMS

NTERMS = NTERMS

Case 2.13 -----

PATH: 3-6 15 7-10 18-19

PREDICATES:

:1 0

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

:7 8

XIN - X(1) .GT. 0

:11 8

XIN - X(2) .LT. 0

:13 10

2 - (NTERMS/2) .GT. 0

:15 19

2 - ((2 - (NTERMS/2) + NTERMS) - 1) .GE. 0

:15 23

CONSISTENCY: OK

OUTPUT:

:1 0

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:15 28

I1 = 2 - (NTERMS/2)

NTERMS = NTERMS

Case 2.14 -----

PATH: 3-8 15 7-8 13-14 59

PREDICATES:

```

:1 8      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .GT. 0
:11 8     XIN = X(2) .EQ. 0
:14 59    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
59      NOTE: Reference to uninitialized variable I1

```

Final Values

```

:14 59    I1 = I1
          NTERMS = NTERMS

```

Case 2.15 -----

PATH: 0-2 15 7-8 15 7 16-24

PREDICATES:

```

:1 2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .GT. 0
:11 8     XIN = X(2) .GT. 0
:17 19    2 - ((1 + 2 - NTERMS + NTERMS) - 1) .LT. 0
:20 22    1 + 2 - NTERMS .LE. 0
:22 28    CONSISTENCY: OK

```

OUTPUT:

Final Values

```

:22 28    I1 = 1
          NTERMS = 2

```

Case 2.16 -----

PATH: 0-3 15 7-8 15 7 16-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .GT. 0
:11 8     XIN = X(2) .GT. 0
:17 19    2 - ((1 + 2 - NTERMS + NTERMS) - 1) .LT. 0
:20 22    1 + 2 - NTERMS .GT. 0
:22 23    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:20 22    I1 = 1 + 2 - NTERMS
          NTERMS = NTERMS

```

Case 2.17 -----

PATH: 0-3 15 7-8 15 7 16-19

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:17 19    2 - ((1 + 2 - NTERMS + NTERMS) - 1) .GE. 0
:17 22    CONSISTENCY: OK
```

OUTPUT:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
```

Final Values

```
:17 22    I1 = 1 + 2 - NTERMS
           NTERMS = NTERMS
```

Case 3 -----

AS IN CASE 1, CHECK FOR NPTS=3.

Case Commands:

```
OUTPUT I1, NTERMS;
SELECT ALL;
CONSISTENCY;
7 ASSIGN NPTS=3;
23 HALT;
```

Case 3.1 -----

PATH: 0-12 18-24

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .LT. 0
:9 10     1 - (NTERMS/2) .LE. 0
:13 19    3 - ((1 + NTERMS) - 1) .LT. 0
:16 22    1 + 3 - NTERMS .LE. 0
:18 28    CONSISTENCY: OK
```

OUTPUT:

Final Values

```
:18 28    I1 = 1
           NTERMS = 3
```

Case 3.2 -----

PATH: 0-12 18-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .LE. 3
:13 19    3 = ((1 + NTERMS) - 1) .LT. 0
:16 22    1 + 3 = NTERMS .GT. 0
:16 22    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:16 28    I1 = 1 + 3 = NTERMS
          NTERMS = NTERMS

```

Case 3.3 -----

PATH: 0-12 13-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .LE. 0
:13 19    3 = ((1 + NTERMS) - 1) .GE. 0
:13 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:13 28    I1 = 1
          NTERMS = NTERMS

```

Case 3.4 -----

PATH: 0-10 18-24

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .GT. 0
:11 19    3 = ((1 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:14 22    1 + 3 = NTERMS .LE. 0
:16 28    CONSISTENCY: OK

```

OUTPUT:

Final Values

```

:16 28    I1 = 1
          NTERMS = 3

```

ORIGINAL PAGE IS
OF POOR QUALITY

Case 3.5 -----

PATH: 3-10 13-22

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 = (NTERMS/2) .GT. 0
:11 19    3 = ((1 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:14 22    1 + 3 = NTERMS .GT. 0
:14 28    CONSISTENCY: OK
```

OUTPUT:

```
:1 2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
```

Final Values

```
:14 28    I1 = 1 + 3 = NTERMS
          NTERMS = NTERMS
```

Case 3.6 -----

PATH: 2-13 13-19

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 3      XIN = X(1) .LT. 0
:9 12     1 = (NTERMS/2) .GT. 0
:11 19    3 = ((1 - (NTERMS/2) + NTERMS) - 1) .GE. 0
:11 28    CONSISTENCY: OK
```

OUTPUT:

```
:1 3      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
```

Final Values

```
:11 28    I1 = 1 - (NTERMS/2)
          NTERMS = NTERMS
```

Case 3.7 -----

PATH: 2-3 13-14 59

PREDICATES:

```
:1 2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .EQ. 0
:13 59    CONSISTENCY: OK
```

OUTPUT:

```
:1 2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
```

59 NOTE: Reference to uninitialized variable I1

Final Values

```
:13 59    I1 = I1
          NTERMS = NTERMS
```

Case 3.8 -----

PATH: 0-3 15 7-12 18-24

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 2     XIN - X(2) .LT. 0
:13 10    2 - (NTERMS/2) .LE. 0
:17 19    3 - ((1 + NTERMS) - 1) .LT. 0
:20 22    1 + 3 - NTERMS .LE. 0
:22 28    CONSISTENCY: OK
```

OUTPUT:

Final Values

```
:22 28      11 = 1
             NTERMS = 3
```

Case 3.9 -----

PATH: 0-3 15 7-12 18-22

PREDICATES:

```
:1 2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .LT. 0
:13 10    2 - (NTERMS/2) .LE. 0
:17 19    3 - ((1 + NTERMS) - 1) .LT. 0
:23 22    1 + 3 - NTERMS .GT. 0
:23 28    CONSISTENCY: OK
```

OUTPUT:

```
:1 8      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
```

Final Values

```
:20 28      11 = 1 + 3 - NTERMS
             NTERMS = NTERMS
```

Case 3.13 -----

PATH: 0-3 15 7-12 18-19

PREDICATES:

```
:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .LT. 0
:13 10    2 - (NTERMS/2) .LE. 0
:17 19    3 - ((1 + NTERMS) - 1) .GE. 0
:17 28    CONSISTENCY: OK
```

OUTPUT:

:1 0

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:17 28

I1 = 1

NTERMS = NTERMS

Case 3.11 -----

PATH: 0-8 15 7-10 18-24

PREDICATES:

:1 0

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

:7 2

XIN - X(1) .GT. 0

:11 8

XIN - X(2) .LT. 0

:13 10

2 - (NTERMS/2) .GT. 0

:15 19

3 - ((2 - NTERMS/2) + NTERMS) - 1) .LT. 0

:18 22

1 + 3 - NTERMS .LE. 2

:23 28

CONSISTENCY: OK

OUTPUT:

Final Values

:23 28

I1 = 1

NTERMS = 3

Case 3.12 -----

PATH: 0-8 15 7-10 18-22

PREDICATES:

:1 0

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

:7 8

XIN - X(1) .GT. 0

:11 3

XIN - X(2) .LT. 0

:13 10

2 - (NTERMS/2) .GT. 0

:15 19

3 - ((2 - (NTERMS/2) + NTERMS) - 1) .LT. 0

:18 22

1 + 3 - NTERMS .GT. 0

:18 28

CONSISTENCY: OK

OUTPUT:

:1 0

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:13 28

I1 = 1 + 3 - NTERMS

NTERMS = NTERMS

Case 3.13 -----

PATH: 0-8 15 7-10 18-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 3      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .LT. 0
:13 10    2 - (NTERMS/2) .GT. 0
:15 19    3 - ((2 - (NTERMS/2) + NTERMS) - 1) .GE. 0
:15 22    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:15 28    I1 = 2 - (NTERMS/2)
          NTERMS = NTERMS

```

Case 3.14 -----

PATH: 0-3 15 7-8 13-14 59

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 3      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .EQ. 0
:14 59    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
59      NOTE: Reference to uninitialized variable I1

```

Final Values

```

:14 59    I1 = I1
          NTERMS = NTERMS

```

Case 3.15 -----

PATH: 0-8 15 7-8 15 7-12 18-24

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .LT. 0
:17 10    3 - (NTERMS/2) .LE. 0
:21 19    3 - ((1 + NTERMS) - 1) .LT. 0
:24 22    1 + 3 - NTERMS .LE. 0
:26 28    CONSISTENCY: OK

```

OUTPUT:

Final Values

```

:26 28    I1 = 1
          NTERMS = 3

```

Case 3.16 -----

PATH: 3-2 15 7-8 15 7-12 13-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .LT. 0
:17 10    3 - (NTERMS/2) .LE. 0
:21 19    3 - ((1 + NTERMS) - 1) .LT. 0
:24 22    1 + 3 - NTERMS .GT. 0
:24 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:24 28    I1 = 1 + 3 - NTERMS
          NTERMS = NTERMS

```

Case 3.17 -----

PATH: 2-8 15 7-8 15 7-12 18-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .LT. 0
:17 13    3 - (NTERMS/2) .LE. 0
:21 19    3 - ((1 + NTERMS) - 1) .GE. 0
:21 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:21 28    I1 = 1
          NTERMS = NTERMS

```

Case 3.18 -----

PATH: 2-8 15 7-8 15 7-10 18-24

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .LT. 0
:17 13    3 - (NTERMS/2) .GT. 0

```

```

:19 19      3 - ((3 - (INTERMS/2) + NTERMS) - 1) .LT. 0
:22 22      1 + 3 - NTERMS .LE. 3
:24 28      CONSISTENCY: OK

```

OUTPUT:

```

Final Values
:24 28      I1 = 1
           NTERMS = 3

```

Case 3.19 -----

PATH: 0-8 15 7-8 15 7-10 10-22

PREDICATES:

```

:1 0        SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8        XIN - X(1) .GT. 0
:11 3       XIN - X(2) .GT. 0
:15 8       XIN - X(3) .LT. 0
:17 10      3 - (INTERMS/2) .GT. 0
:19 19      3 - ((3 - (INTERMS/2) + NTERMS) - 1) .LT. 0
:22 22      1 + 3 - NTERMS .GT. 0
:22 28      CONSISTENCY: OK

```

OUTPUT:

```

:1 0        SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values
:22 28      I1 = 1 + 3 - NTERMS
           NTERMS = NTERMS

```

Case 3.20 -----

PATH: 0-8 15 7-8 15 7-10 10-19

PREDICATES:

```

:1 0        SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8        XIN - X(1) .GT. 0
:11 3       XIN - X(2) .GT. 0
:15 8       XIN - X(3) .LT. 0
:17 10      3 - (INTERMS/2) .GT. 0
:19 19      3 - ((3 - (INTERMS/2) + NTERMS) - 1) .GE. 0
:19 28      CONSISTENCY: OK

```

OUTPUT:

```

:1 0        SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values
:19 28      I1 = 3 - (INTERMS/2)
           NTERMS = NTERMS

```

ORIGINAL PAGE IS
OF POOR QUALITY

Case 3.21 -----

PATH: 0-8 15 7-8 15 7-8 13-14 59

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .EQ. 0
:18 59    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
59      NOTE: Reference to uninitialized variable I1

```

Final Values

```

:18 59    I1 = I1
          NTERMS = NTERMS

```

Case 3.22 -----

PATH: 0-8 15 7-8 15 7-8 15 7 16-24

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .GT. 0
:21 19    3 - ((1 + 3 - NTERMS + NTERMS) - 1) .LT. 0
:24 22    1 + 3 - NTERMS .LE. 0
:26 28    CONSISTENCY: OK

```

OUTPUT:

Final Values

```

:26 28    I1 = 1
          NTERMS = 3

```

Case 3.23 -----

PATH: 0-8 15 7-8 15 7-8 15 7 16-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .GT. 0
:21 19    3 - ((1 + 3 - NTERMS + NTERMS) - 1) .LT. 0
:24 22    1 + 3 - NTERMS .GT. 0
:24 28    CONSISTENCY: OK

```

OUTPUT:
:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values
:24 28 I1 = 1 + 3 - NTERMS
NTERMS = NTERMS

Case 3.24 -----

PATH: 3-8 15 7-8 15 7-8 15 7 16-19

PREDICATES:
:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .GT. 0
:15 8 XIN - X(3) .GT. 0
:21 19 3 - ((1 + 3 - NTERMS + NTERMS) - 1) .GE. 0
:21 28 CONSISTENCY: OK

OUTPUT:
:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values
:21 28 I1 = 1 + 3 - NTERMS
NTERMS = NTERMS

Case 4 -----

AS IN CASE 1, CHECK FOR NPTS=4.

Case Commands:

OUTPUT I1, NTERMS;
SELECT ALL;
CONSISTENCY;
7 ASSIGN NPTS=4;
28 HALT;

Case 4.1 -----

PATH: 8-12 18-24

PREDICATES:
:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .LT. 0
:9 10 1 - (NTERMS/2) .LE. 0
:13 19 4 - ((1 + NTERMS) - 1) .LT. 0
:16 22 1 + 4 - NTERMS .LE. 0
:18 28 CONSISTENCY: OK

OUTPUT:

Final Values
:13 28

I1 = 1
NTERMS = 4

Case 4.2 -----

PATH: 0-12 18-22

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .LT. 0
:9 10 1 - (NTERMS/2) .LE. 0
:13 19 4 - ((1 + NTERMS) - 1) .LT. 0
:16 22 1 + 4 - NTERMS .GT. 0
:16 28 CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values
:16 28

I1 = 1 + 4 - NTERMS
NTERMS = NTERMS

Case 4.3 -----

PATH: 0-12 18-19

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .LT. 0
:9 10 1 - (NTERMS/2) .LE. 0
:13 19 4 - ((1 + NTERMS) - 1) .GE. 0
:13 28 CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values
:13 28

I1 = 1
NTERMS = NTERMS

Case 4.4 -----

PATH: 0-13 18-24

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .LT. 0

```

:9 10      1 - (NTERMS/2) .GT. 0
:11 19     4 - ((1 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:14 22     1 + 4 - NTERMS .LE. 0
:16 28     CONSISTENCY: OK

```

OUTPUT:

```

Final Values
:16 28      I1 = 1
           NTERMS = 4

```

Case 4.5 -----

PATH: 8-10 18-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 - (NTERMS/2) .GT. 0
:11 19    4 - ((1 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:14 22    1 + 4 - NTERMS .GT. 0
:14 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values
:14 28     I1 = 1 + 4 - NTERMS
           NTERMS = NTERMS

```

Case 4.6 -----

PATH: 0-10 18-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .LT. 0
:9 10     1 - (NTERMS/2) .GT. 0
:11 19    4 - ((1 - (NTERMS/2) + NTERMS) - 1) .GE. 0
:11 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values
:11 28     I1 = 1 - (NTERMS/2)
           NTERMS = NTERMS

```

ORIGINAL PAGE IS
OF POOR QUALITY

Case 4.7 -----

PATH: 0-8 13-14 59

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .EQ. 0
:10 59 CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
59 NOTE: Reference to uninitialized variable I1

Final Values

:10 59 I1 = I1
NTERMS = NTERMS

Case 4.8 -----

PATH: 0-8 15 7-12 18-24

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .LT. 0
:13 10 2 - (NTERMS/2) .LE. 0
:17 19 4 - ((1 + NTERMS) - 1) .LT. 0
:23 22 1 + 4 - NTERMS .LE. 0
:22 23 CONSISTENCY: OK

OUTPUT:

Final Values

:22 23 I1 = 1
NTERMS = 4

Case 4.9 -----

PATH: 0-8 15 7-12 18-22

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .LT. 0
:13 10 2 - (NTERMS/2) .LE. 0
:17 19 4 - ((1 + NTERMS) - 1) .LT. 0
:20 22 1 + 4 - NTERMS .GT. 0
:23 28 CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:23 23 I1 = 1 + 4 - NTERMS

NTERMS = ITERMS

Case 4.10 -----

PATH: 0-3 15 7-12 18-19

PREDICATES:
:1 2 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .LT. 0
:13 10 2 - (NTERMS/2) .LE. 0
:17 19 4 - ((1 + NTERMS) - 1) .GE. 0
:17 22 CONSISTENCY: OK

OUTPUT:
:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values
:17 22 I1 = 1
NTERMS = NTERMS

Case 4.11 -----

PATH: 0-3 15 7-10 18-24

PREDICATES:
:1 2 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .LT. 0
:13 10 2 - (NTERMS/2) .GT. 0
:15 19 4 - ((2 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:18 22 1 + 4 - NTERMS .LE. 0
:20 22 CONSISTENCY: OK

OUTPUT:

Final Values
:20 22 I1 = 1
NTERMS = 4

Case 4.12 -----

PATH: 0-3 15 7-13 18-22

PREDICATES:
:1 2 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .LT. 0
:13 10 2 - (NTERMS/2) .GT. 0
:15 19 4 - ((2 - (NTERMS/2) + NTERMS) - 1) .LT. 0

:18 22 1 + 4 - NTERMS .GT. 0
:18 23 CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:18 23 I1 = 1 + 4 - NTERMS
NTERMS = NTERMS

Case 4.13 -----

PATH: 0-8 15 7-10 12-19

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .LT. 0
:12 10 2 - (NTERMS/2) .GT. 0
:15 19 4 - ((2 - (NTERMS/2) + NTERMS) - 1) .GE. 0
:15 22 CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:15 22 I1 = 2 - (NTERMS/2)
NTERMS = NTERMS

Case 4.14 -----

PATH: 0-8 15 7-8 13-14 59

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .EQ. 0
:14 59 CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

59 NOTE: Reference to uninitialized variable I1

Final Values

:14 59 I1 = I1
NTERMS = NTERMS

Case 4.15 -----

PATH: 0-8 15 7-8 15 7-12 18-24

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 3     XIN - X(2) .GT. 0
:15 3     XIN - X(3) .LT. 0
:17 10    3 - (NTERMS/2) .LE. 0
:21 19    4 - ((1 + NTERMS) - 1) .LT. 0
:24 22    1 + 4 - NTERMS .LE. 0
:26 28    CONSISTENCY: OK

```

OUTPUT:

Final Values

```

:26 28    I1 = 1
          NTERMS = 4

```

Case 4.16 -----

PATH: 0-8 15 7-8 15 7-12 18-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .LT. 0
:17 10    3 - (NTERMS/2) .LE. 0
:21 19    4 - ((1 + NTERMS) - 1) .LT. 0
:24 22    1 + 4 - NTERMS .GT. 0
:24 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:24 28    I1 = 1 + 4 - NTERMS
          NTERMS = NTERMS

```

Case 4.17 -----

PATH: 2-8 15 7-8 15 7-12 18-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .LT. 0
:17 10    3 - (NTERMS/2) .LE. 0
:21 19    4 - ((1 + NTERMS) - 1) .GE. 0
:21 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

:21 28

I1 = 1

NTERMS = NTERMS

Case 4.18 -----

PATH: 0-8 15 7-8 15 7-10 18-24

PREDICATES:

:1 0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 3	XIN - X(1) .GT. 0
:11 3	XIN - X(2) .GT. 0
:15 3	XIN - X(3) .LT. 3
:17 10	3 - (NTERMS/2) .GT. 0
:19 19	4 - ((3 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:22 22	1 + 4 - NTERMS .LE. 0
:24 28	CONSISTENCY: OK

OUTPUT:

Final Values

:24 28

I1 = 1

NTERMS = 4

Case 4.19 -----

PATH: 0-8 15 7-8 15 7-10 18-22

PREDICATES:

:1 0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8	XIN - X(1) .GT. 0
:11 8	XIN - X(2) .GT. 0
:15 8	XIN - X(3) .LT. 0
:17 10	3 - (NTERMS/2) .GT. 0
:19 19	4 - ((3 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:22 22	1 + 4 - NTERMS .GT. 0
:22 28	CONSISTENCY: OK

OUTPUT:

:1 0

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:22 28

I1 = 1 + 4 - NTERMS

NTERMS = NTERMS

Case 4.20 -----

PATH: 0-8 15 7-8 15 7-10 18-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .LT. 0
:17 10    3 - (NTERMS/2) .GT. 0
:19 19    4 - ((3 - (NTERMS/2) + NTERMS) - 1) .GE. 0
:19 20    CONSISTENCY: OK

```

OUTPUT:

```

:1 2      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:19 20    I1 = 3 - (NTERMS/2)
          NTERMS = NTERMS

```

Case 4.21 -----

PATH: 3-8 15 7-8 15 7-8 13-14 59

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .EQ. 0
:18 59    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
59      NOTE: Reference to uninitialized variable I1

```

Final Values

```

:18 59    I1 = I1
          NTERMS = NTERMS

```

Case 4.22 -----

PATH: 0-8 15 7-8 15 7-8 15 7-12 18-24

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .GT. 0
:19 8     XIN - X(4) .LT. 0
:21 10    4 - (NTERMS/2) .LE. 0
:25 19    4 - ((1 + NTERMS) - 1) .LT. 0
:28 22    1 + 4 - NTERMS .LE. 0
:30 28    CONSISTENCY: OK

```

OUTPUT:

Final Values

ORIGINAL PAGE IS
OF POOR QUALITY

:30 23

I1 = 1
NTERMS = 4

Case 4.23 -----

PATH: 3-8 15 7-8 15 7-8 15 7-12 18-22

PREDICATES:

:1 0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8	XIN - X(1) .GT. 0
:11 8	XIN - X(2) .GT. 0
:15 8	XIN - X(3) .GT. 0
:19 8	XIN - X(4) .LT. 0
:21 10	4 - (NTERMS/2) .LE. 0
:25 19	4 - ((1 + NTERMS) - 1) .LT. 0
:28 22	1 + 4 - NTERMS .GT. 0
:28 22	CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:28 23 I1 = 1 + 4 - NTERMS
NTERMS = NTERMS

Case 4.24 -----

PATH: 0-8 15 7-8 15 7-8 15 7-12 18-19

PREDICATES:

:1 0	SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8	XIN - X(1) .GT. 0
:11 8	XIN - X(2) .GT. 0
:15 8	XIN - X(3) .GT. 0
:19 8	XIN - X(4) .LT. 0
:21 10	4 - (NTERMS/2) .LE. 0
:25 19	4 - ((1 + NTERMS) - 1) .GE. 0
:25 28	CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:25 23 I1 = 1
NTERMS = NTERMS

Case 4.25 -----

PATH: 0-8 15 7-8 15 7-8 15 7-12 18-24

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .GT. 0
:19 8     XIN - X(4) .LT. 0
:21 10    4 - (NTERMS/2) .GT. 0
:23 19    4 - ((4 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:26 22    1 + 4 - NTERMS .LE. 0
:28 28    CONSISTENCY: OK

```

OUTPUT:

Final Values

```

:28 28    I1 = 1
          NTERMS = 4

```

Case 4.26 -----

PATH: 0-8 15 7-8 15 7-8 15 7-10 18-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .GT. 0
:19 8     XIN - X(4) .LT. 0
:21 10    4 - (NTERMS/2) .GT. 0
:23 19    4 - ((4 - (NTERMS/2) + NTERMS) - 1) .LT. 0
:26 22    1 + 4 - NTERMS .GT. 0
:26 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:26 28    I1 = 1 + 4 - NTERMS
          NTERMS = NTERMS

```

Case 4.27 -----

PATH: 0-8 15 7-2 15 7-8 15 7-10 18-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN - X(1) .GT. 0
:11 8     XIN - X(2) .GT. 0
:15 8     XIN - X(3) .GT. 0
:19 8     XIN - X(4) .LT. 0
:21 10    4 - (NTERMS/2) .GT. 0
:23 19    4 - ((4 - (NTERMS/2) + NTERMS) - 1) .GE. 0
:23 28    CONSISTENCY: OK

```

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

Final Values

:23 28 I1 = 4 - (NTERMS/2)
NTERMS = NTERMS

Case 4.23 -----

PATH: 0-8 15 7-8 15 7-8 15 7-2 13-14 59

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .GT. 0
:15 8 XIN - X(3) .GT. 0
:19 8 XIN - X(4) .EQ. 0
:22 59 CONSISTENCY: OK

OUTPUT:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

59 NOTE: Reference to uninitialized variable I1

Final Values

:22 59 I1 = I1
NTERMS = NTERMS

Case 4.29 -----

PATH: 0-8 15 7-8 15 7-8 15 7-2 15 7 16-24

PREDICATES:

:1 0 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8 XIN - X(1) .GT. 0
:11 8 XIN - X(2) .GT. 0
:15 8 XIN - X(3) .GT. 0
:19 8 XIN - X(4) .GT. 0
:25 19 4 - ((1 + 4 - NTERMS + NTERMS) - 1) .LT. 0
:28 22 1 + 4 - NTERMS .LE. 0
:32 28 CONSISTENCY: OK

OUTPUT:

Final Values

:32 28 I1 = 1
NTERMS = 4

Case 4.30 -----

PATH: 0-8 15 7-8 15 7-8 15 7-8 15 7 16-22

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .GT. 0
:11 8     XIN = X(2) .GT. 0
:15 8     XIN = X(3) .GT. 0
:19 8     XIN = X(4) .GT. 0
:25 19    4 - ((1 + 4 - NTERMS + NTERMS) - 1) .LT. 0
:28 22    1 + 4 - NTERMS .GT. 0
:28 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:28 28    I1 = 1 + 4 - NTERMS
          NTERMS = NTERMS

```

Case 4.31 -----

PATH: 0-8 15 7-8 15 7-8 15 7-8 15 7 16-19

PREDICATES:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:7 8      XIN = X(1) .GT. 0
:11 8     XIN = X(2) .GT. 0
:15 8     XIN = X(3) .GT. 0
:19 8     XIN = X(4) .GT. 0
:25 19    4 - ((1 + 4 - NTERMS + NTERMS) - 1) .GE. 0
:25 28    CONSISTENCY: OK

```

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)

```

Final Values

```

:25 28    I1 = 1 + 4 - NTERMS
          NTERMS = NTERMS

```

Case 5 -----

ANALYSIS OF THE PART OF THE CODE THAT SETS UP DELTA (DELTA)
AND DELTA(1). SET NTERMS TO 1.

Case Commands:

```

OUTPUT DELTA, DELTA;
7 SKIP 28;
28 ASSIGN NTERMS=1;
28 ASSIGN I1="I1";
36 HALT;

```

Case 5.1 -----

PATH: 2-3 23-32.1 33

PREDICATES:

OUTPUT:

:1 3
:6 28

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
** ASSIGN I1 = I1

Final Values

:14 33

DELTA(X) = (XIN - X(I1))/(X(1 + I1) - X(I1))

DELTA(1) = (X((1 + I1) - 1) - X(I1))/(X(1 +
I1) - X(I1))

Case 6 =====

AS IN CASE 5, NTERMS=4.

Case Commands:

OUTPUT DELTA(X), DELTA;

7 SKIP 28;

28 ASSIGN NTERMS=4;

28 ASSIGN I1="I1";

36 HALT;

Case 6.1 -----

PATH: 2-3 23-32.1 33-32.1 33-32.1 33-32.1 33

PREDICATES:

OUTPUT:

:1 3
:6 28

SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
** ASSIGN I1 = I1

Final Values

:29 33

DELTA(X) = (XIN - X(I1))/(X(1 + I1) - X(I1))

DELTA(4) = (X((4 + I1) - 1) - X(I1))/(X(1 +
I1) - X(I1))

DELTA(3) = (X((3 + I1) - 1) - X(I1))/(X(1 +
I1) - X(I1))

DELTA(2) = (X((2 + I1) - 1) - X(I1))/(X(1 +
I1) - X(I1))

DELTA(1) = (X((1 + I1) - 1) - X(I1))/(X(1 +
I1) - X(I1))

Case 7 =====

ANALYSIS: THE PART OF THE CODE USED TO GENERATE THE COEFFICIENTS.
 POLYNOMIAL: DEGREE 1. NTERMS=2.
 IN THIS CASE WE GET THE RECURSIVE FORMULA TO COMPARE WITH THE
 DOCUMENTED ONE BY ASSIGNING A TO SYMBOLIC A.

Case Commands:

```

7 SKIP 36;
36 ASSIGN I1="I1";
36 OUTPUT I1;
37 ASSIGN A="A";
36 ASSIGN NTERMS=2;
36 ASSIGN DELTA="DELTA";
46 OUTPUT A;
47 ASSIGN A="A";
51 HALT;

```

Case 7.1 -----

PATH: 8- 36-45 42 46-47 37

PREDICATES:

OUTPUT:

```

:1 SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:5 ** ASSIGN I1 = 11
:7 ** ASSIGN DELTA = DELTA(I1)
:8 A(1) = Y(I1)
:9 ** ASSIGN A = A(I1)
:21 A(2) = -A(1)/(DELTA(2) - DELTA(1)) + (Y(1 +
      11)/(DELTA(2) - DELTA(1)))

```

Case 8 -----

AS IN CASE 7 EXCEPT POLYNOMIAL OF DEGREE 2. NTERMS=3.

Case Commands:

```

7 SKIP 36;
36 ASSIGN I1="I1";
36 OUTPUT I1;
37 ASSIGN A="A";
36 ASSIGN NTERMS=3;
36 ASSIGN DELTA="DELTA";
46 OUTPUT A;
47 ASSIGN A="A";
51 HALT;

```

ORIGINAL PAGE IS
 OF POOR QUALITY

Case 8.1 -----

PATH: 0-3 38-45 42 46-47 37-45 42-45 42 46-47 37

PREDICATES:

OUTPUT:

```

:1 0      SUBROUTINE INTERP(X, Y, NPTS, NTERMS, XIN, YOUT)
:5 38     ** ASSIGN I1 = I1
:7 38     ** ASSIGN DELTA = DELTA(SI)
:8 36     A(1) = Y(I1)
:9 37     ** ASSIGN A = A(SI):9
:21 43    A(2) = -A(1):9/(DELTA(2) - DELTA(1)) + (Y(1 +
          11)/(DELTA(2) - DELTA(1)))
:22 47    ** ASSIGN A = A(SI):22
:41 46    A(3) = -A(2):22/(DELTA(3) - DELTA(2)) - (A(1):22/((
          DELTA(2) - DELTA(2))* (DELTA(3) - DELTA(1))))
          + (Y(2 + 11)/((DELTA(3) - DELTA(2))* (
          DELTA(3) - DELTA(1))))
```

Case 9 =====

ANALYSIS OF PART OF PROGRAM THAT USES COEFFICIENTS FROM CASES 7 AND 8
TO CALCULATE OUTPUT VALUE OF YOUT. AS IN CASE 7, POLYNOMIAL OF
DEGREE 1. NTERMS=2.

Case Commands:

```

OUTPUT YOUT;
7 SKIP 51;
51 ASSIGN NTERMS=2;
51 ASSIGN A="A";
51 ASSIGN DELTAX="DELTAX";
51 ASSIGN DELTA="DELTA";
```

Case 9.1 -----

PATH: 2-3 51-56 55 57 52 53-59

PREDICATES:

OUTPUT:

```

:6 51     ** ASSIGN A = A(SI)
:7 51     ** ASSIGN DELTAX = DELTAX
:8 51     ** ASSIGN DELTA = DELTA(SI)
```

Final Values

```

:21 59    YOUT = A(1) + A(2)*(DELTAX - DELTA(1))
```

Case 10 =====

AS IN CASE 9 EXCEPT POLYNOMIAL OF DEGREE 2. NTERMS=3.

Case Commands:

OUTPUT YOUT;
7 SKIP S1;
S1 ASSIGN NTERMS=3;
S1 ASSIGN A="A";
S1 ASSIGN DELTAX="DELTAX";
S1 ASSIGN DELTA="DELTA";

Case 13.1 -----

PATH: 0-3 51-53 55 57 52-53 55-56 55 57 52 53-59

PREDICATES:

OUTPUT:

:6	S1	** ASSIGN A = A(S1)
:7	S1	** ASSIGN DELTAX = DELTAX
:8	S1	** ASSIGN DELTA = DELTA(S1)

Final Values

:33 59

YOUT = A(1) + A(2)*(DELTAX - DELTA(1)) + A(3)*(
DELTAX - DELTA(1))*(DELTAX - DELTA(2))

MAIN

```

1. C FL. DESCRIPTION OF METHOD USED IN THIS PROGRAM, SEE "THE OR
2. C TRANSFORMATION" BY FRANCIS, COMPUTER JOURNAL, VOL. 4 PP. 265,332
3. C START HERE TO PRODUCE UPPER HESSENBURG MATRIX
4. C DIMENSION A(NMAX,NMAX),INT(1),VEC(1),D(NMAX,NMAX),EIGVEC(NMAX,NMAX)
5. C
6. N=NM
7. K=N-2
8. DO1J=1,N
9. K1=J+1
10. K2=J+2
11. GREAT= ABS(A(K1,J))
12. MAX=K1
13. DO2I=K2,N
14. AB= ABS(A(I,J))
15. IF (GREAT.GE.AB) GO TO 2
16. 3 GREAT=AB
17. MAX=I
18. 2 CONTINUE
19. C INT(1) GIVES INFORMATION ABOUT INTERCHANGES
20. INT(1)=MAX
21. IF (MAX.EQ.K1) GO TO 5
22. 4 DO4L=J,N
23. E=A(K1,L)
24. A(K1,L)=A(MAX,L)
25. 6 A(MAX,L)=E
26. DO7L=1,N
27. E=A(L,K1)
28. A(L,K1)=A(L,MAX)
29. 7 A(L,MAX)=E
30. C INTERCHANGE DONE
31. 5 DO8I=K2,N
32. IF (A(I,J).EQ.0.) GO TO 8
33. A(I,J)=A(I,J)/A(K1,J)
34. DO11M=K1,N
35. 11 A(I,M)=A(I,M)-A(I,J)*A(K1,M)
36. 8 CONTINUE
37. C LEFT MULT. DONE
38. DO 2000 M=K2,N
39. IF (A(M,J).EQ.0.) GO TO 2000
40. 10 DO 2001 I=1,N
41. 2001 A(I,K1)=A(I,K1)-A(I,M)*A(M,J)
42. 2000 CONTINUE
43. 1 CONTINUE
44. C RIGHT MULT. DONE
45. C HESSENBURG MATRIX NOW FORMED AS A(I,J). MULTIPLIERS STORED WITH IT
46. DO 105 I=1,N
47. DO 105 J=1,N
48. 105 D(I,J)=A(I,J)
49. C PUT APPROPRIATE ZEROS WHERE MULTIPLIERS WERE STORED
50. DO 106 I=3,N
51. L=I-2
52. DO 106 J=1,L
53. 106 A(I,J)=0.0
54. K1=N+1
55. R=0.0
56. C=0.0
57. DO33I=1,N
58. R=R+ ABS(A(I,I))
59. 33 C=C+ ABS(A(I,N))
60. DO34I=2,N
61. RS=0.0
62. CS=0.0
63. L=I-1
64. DO35J=L,N
65. RS=RS+ ABS(A(I,J))
66. K1J=K1-J
67. K11=K1-I
68. 35 CS=CS+ ABS(A(K1J,K11))
69. IF (RS.LE.N) GO TO 36
70. 37 R=RS
71. 36 IF (CS.LE.C) GO TO 34
72. 44 C=CS

```

29 MAR 76 G.04-01

MAIN-1

ORIGINAL PAGE IS
OF POOR QUALITY

1J

```

73. 34 CONTINUE
74. IJK=1
75. IF (N.GE.C) GO TO 54
76. C DELTA IS CRITERION FOR STOPPING AN ITERATION
77. 53 DELTA=R*1.0E-11
78. C DEL IS ARBITRARY SHIFT
79. DEL=R/17.1029
80. GO TO 153

81. 54 DELTA=C*1.0E-11
82. DEL=C/17.1029
83. C DELTA COMPUTED
84. C PRODUCE A-PI, ITERATE ON THIS TO BREAK UP EIGENVALUES OF EQUAL MODULUS

85. 153 GO 107 (I=1,N
86. 107 A(I,1)=A(I,1)-DEL

87. 55 IF (N.NE.1) GO TO 48
88. C SUBROUTINE CALCEV CALCULATES EIGENVALUES OF 1X1 OR 2X2 MATRICES.
89. C PI IS ADDED BACK IN CALCEV
90. 47 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
91. GO TO 74

92. 48 IF (N.NE.2) GO TO 80
93. 49 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
94. GO TO 74
95. C ON TRANSFORMATION STARTS HERE
96. C INITIALIZE FOR DETERMINING SUBSEQUENT SHIFTS

97. 80 AR1=0.0
98. AR3=0.0
99. AR4=0.0
100. C RE-ENTER AFTER REALIT OR COMPT SUBROUTINES CALLED
101. C SELECT FIRST (I=ZERO) SUB-DIAGONAL ELEMENT FROM BOTTOM

102. 61 L=N
103. D056 I=2,N
104. N21=N+2-I
105. IF (ABS(A(N21,N21-1)).LT.DELTA) GO TO 57
106. 56 L=L-1
107. C VALUE OF L SELECTED

108. 57 IF (L.NE.N) GO TO 59
109. 58 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
110. N=N-1
111. GO TO 48

112. 59 IF ((L-N+1).NE.0) GO TO 75
113. 60 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
114. N=N-2
115. GO TO 55
116. C DETERMINE EIGENVALUES OF 2X2 MATRIX -- ROW 1 = A(N-1,N-1) A(N-1,N)
117. C ROW 2 = A(N,N-1) A(N,N)
118. C AND BASE SHIFT ON THESE

119. 75 B=A(N-1,N-1)*A(N,N)
120. C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
121. C1=A(N-1,N-1)-A(N,N)
122. DISC=C1*4.0+A(N,N-1)*A(N-1,N)
123. IF (DISC.GE.0.) GO TO 63
124. 62 R1=0.5*B
125. R2=.5* SQRT( ABS(DISC))
126. ANUM=(R1-AR3)*(R1-AR3)*(R2-AR4)*(R2-AR4)
127. DENOM=AR3*AR3*AR4*AR4
128. IF (DENOM.NE.0.) GO TO 200
129. 201 RHO=0.0
130. SIGMA=0.0
131. GO TO 203

132. 200 RHO=C
133. SIGMA=B
134. 203 AR3=R1
135. AR4=R2
136. CALL COMPT(A,NMAX,L,N,RHO,SIGMA)
137. GO TO 61

139. 63 IF (B.GE.0.) GO TO 65
140. 64 R1=.5*(B- SQRT(DISC))
141. GO TO 66

```

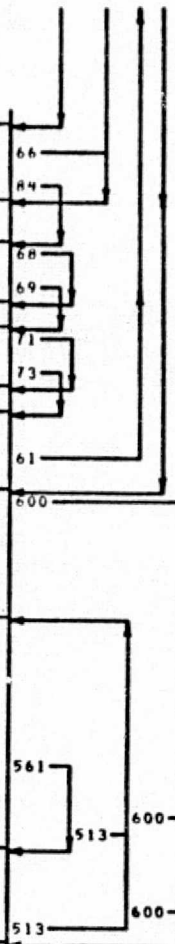
MAIN-2

29 MAR 76 8.04-01

```

142. 65 R1=.5*(B+SQRT(DISC))
143. IF (R1.NE.0.) GO TO 66
144. 63 R2=0.0
145. GO TO 64
146. 66 R2=C/R1
147. C ROOTS OF 2*2 MATRIX FOUND
148. 64 IF (ABS(A(N,N)-R1).GE. ABS(A(N,N)-R2))GO TO 68
149. 67 AR2=R1
150. GO TO 69
151. 68 AR2=R2
152. 69 IF (AR1.NE.0.) GO TO 71
153. 72 SHIFT=0.0
154. GO TO 73
155. 71 SHIFT=AR2
156. 73 AR1=AR2
157. CALL REALIT(A,NMAX,L,N,SHIFT,CC,SS)
158. GO TO 61
159. C ARE VECTORS NEEDED
160. 76 IF (NVEC.EQ.0) GO TO 600
161. C INVERSE ITERATION IS USED FOR EIGENVECTORS
162. 550 SMALL=DELTA*1.0E-30
163. NUM=0
164. J1=1
165. NM=NM+NM
166. C C1 IS REAL PART OF EIGENVALUE
167. 513 C1=VEC(J1)
168. C C2 IS COMPLEX PART OF EIGENVALUE
169. C2=VEC(J1+1)
170. NUM=NUM+1
171. C REPLACE HESSENBURG MATRIX AND MULTIPLIERS
172. DO 570 I=1,NM
173. DO 570 J=1,NM
174. 570 A(I,J)=D(I,J)
175. IF (C2.NE.0.) GO TO 561
176. CALL RVEC(A,NMAX,INT,C1,SMALL,NM,NUM,EIGVEC,NVEC,CC,SS)
177. J1=J1+2
178. IF(NVEC.EQ.NUM)GO TO 600
179. GO TO 513
180. 561 CALL IMVEC(A,NMAX,INT,C1,C2,SMALL,NM,NUM,EIGVEC,NVEC,CC,SS,Z,BB)
181. C NEXT STATEMENT IGNORES CONJUGATE COMPLEX EIGENVALUES
182. J1=J1+4
183. IF(NVEC.EQ.NUM)GO TO 600
184. GO TO 513
185. 600 RETURN
186. END

```



CALCEV

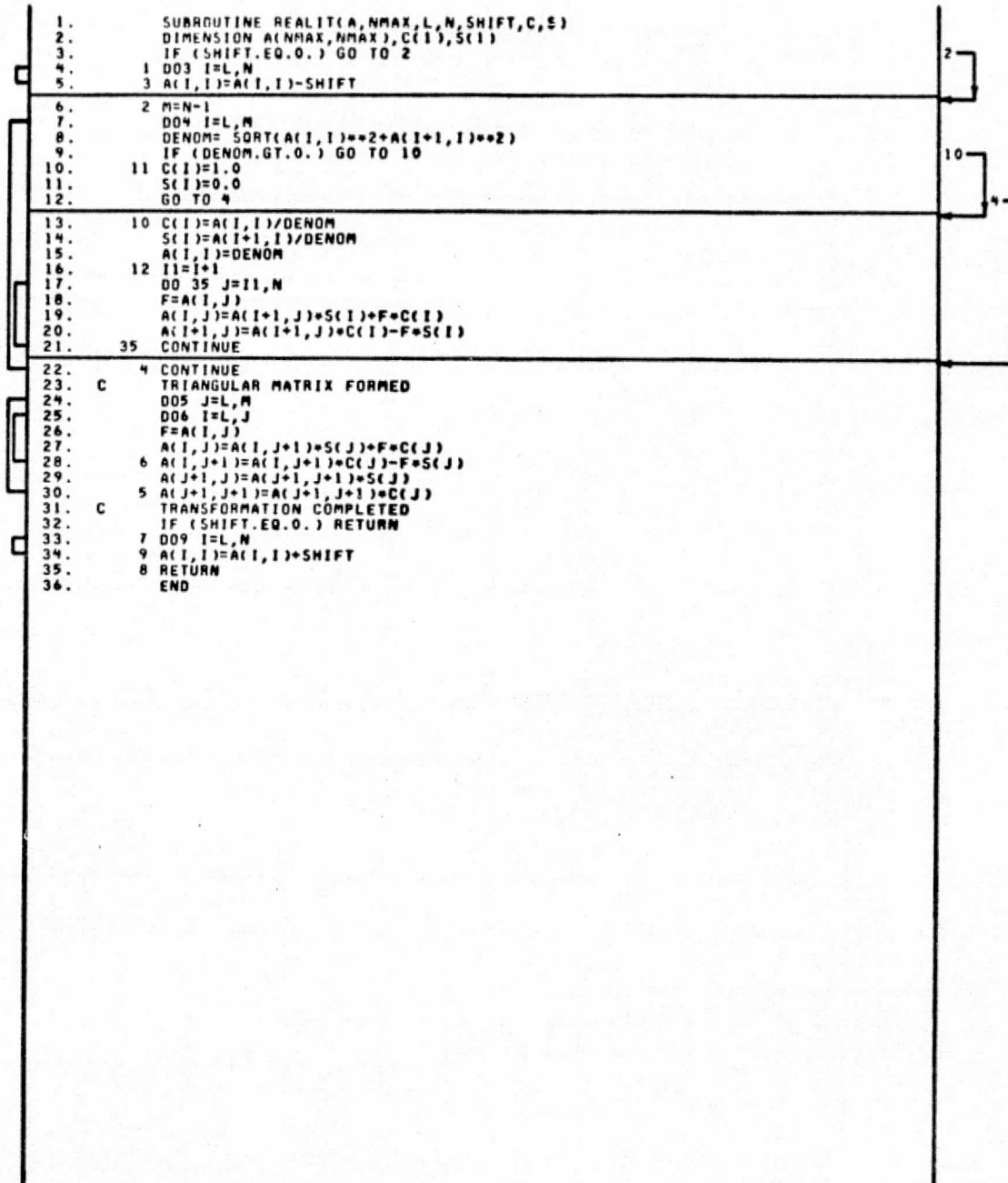
```

1.      SUBROUTINE CALCEV(A,NMAX,DEL,VEC,IJK,N,M)
2.      DIMENSION A(NMAX,NMAX),VEC(1)
3.      IF (M.NE.1) GO TO 2
4.      1 A1=0.0
5.      A(N,N)=A(N,N)+DEL
6.      C STORE EIGENVALUES IN VEC
7.      VEC(IJK)=A(N,N)
8.      VEC(IJK+1)=A1
9.      IJK=IJK+2
10.     RETURN
11.     2 B=A(N-1,N-1)+A(N,N)
12.     C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
13.     C1=A(N-1,N-1)-A(N,N)
14.     DISC=C1+C1+4.0*A(N,N-1)*A(N-1,N)
15.     IF (DISC.LT.0.) GO TO 9
16.     4 A1=0.0
17.     IF (B.GE.0.) GO TO 6
18.     R1=0.5*(B-SQRT(DISC))
19.     GO TO 7
20.     6 R1=0.5*(B+SQRT(DISC))
21.     IF (R1.NE.0.) GO TO 7
22.     10 R2=0.
23.     GO TO 11
24.     7 R2=C/R1
25.     11 R1=R1+DEL
26.     R2=R2+DEL
27.     VEC(IJK)=R2
28.     VEC(IJK+1)=A1
29.     VEC(IJK+2)=R1
30.     VEC(IJK+3)=A1
31.     GO TO 8
32.     9 R1=0.5*B
33.     A1=-.5*SQRT(ABS(DISC))
34.     R1=R1+DEL
35.     VEC(IJK)=R1
36.     VEC(IJK+1)=-A1
37.     VEC(IJK+2)=R1
38.     VEC(IJK+3)=A1
39.     8 IJK=IJK+4
40.     RETURN
41.     END

```



REALIT

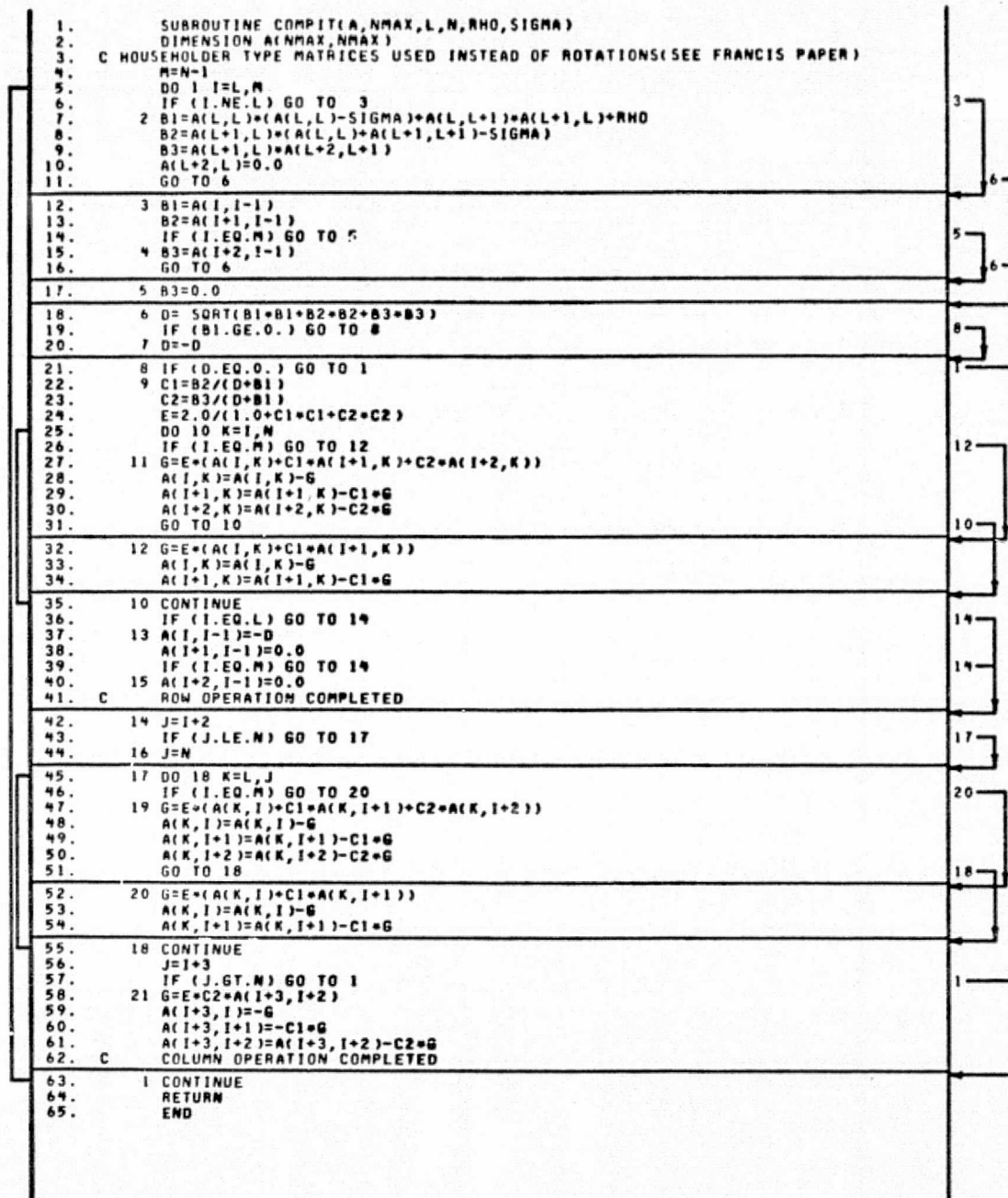


29 MAR 76 G.04-01

REALIT-1

ORIGINAL PAGE IS
OF POOR QUALITY

COMFIT



29 MAR 76 G.04-01

COMFIT-1

RVEC

```

1.  SUBROUTINE RVEC(A,NMAX,INT,C1,SMALL,N,NUM,EIGVEC,NVEC,INTER,Y)
2.  DIMENSION A(NMAX,NMAX),INTER(1),Y(1),INT(1),EIGVEC(NMAX,NMAX)
3.  DO1 I=1,N
4.  1 A(I,1)=A(I,1)-C1
5.  DO2 I=2,N
6.  IF(ABS(A(I-1,I-1)).GE. ABS(A(I,I)))GO TO 4
7.  3 INTER(I)=I
8.  J=I-1
9.  DO5 K=J,N
10. B=A(I-1,K)
11. A(I-1,K)=A(I,K)
12. 5 A(I,K)=B
13. GO TO 6
14.  INTER(I)=I-1
15.  IF (A(I,I-1).EQ.0.) GO TO 2
16.  C=-A(I,I-1)/A(I-1,I-1)
17.  A(I,I-1)=C
18.  DO 8 K=1,N
19.  8 A(I,K)=A(I,K)+C*A(I-1,K)
20.  2 CONTINUE
21.  C TRIANGULAR MATRIX FORMED
22.  DO9 I=1,N
23.  IF(ABS(A(I,I)).GE.SMALL)GO TO 9
24.  10 A(I,I)=SMALL
25.  9 CONTINUE
26.  C NORMALISED ((EIGENVECTOR)) OF HESSENBURG MATRIX OBTAINED FOR RMS = 1
27.  DO 5555 I=1,N
28.  5555 Y(I)=1.0
29.  DO 8888 JJJJ=1,4
30.  7777 Y(N)=Y(N)/A(N,N)
31.  B=Y(N)
32.  DO11 I=2,N
33.  E=0.0
34.  M=N+1-I
35.  DO12 J=2,I
36.  L=M+J-I
37.  12 E=E+Y(L)*A(M,L)
38.  Y(M)=(Y(M)-E)/A(M,M)
39.  IF(ABS(B).GE. ABS(Y(M)))GO TO 11
40.  13 B=Y(M)
41.  11 CONTINUE
42.  DO14 I=1,N
43.  14 Y(I)=Y(I)/B
44.  IF(JJJJ.EQ.4)GO TO 6666
45.  DO15 I=2,N
46.  IF (INTER(I).NE.I) GO TO 15
47.  16 B=Y(I)
48.  Y(I)=Y(I-1)
49.  Y(I-1)=B
50.  15 Y(I)=Y(I)+A(I,I-1)*Y(I-1)
51.  8888 CONTINUE
52.  C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED (NOT NORMALIZED)
53.  6666 M=N-1
54.  DO21 I=2,M
55.  J=M+1-I
56.  L=J+1
57.  DO20 K=L,N
58.  20 Y(K)=Y(K)-Y(J)*A(K,J-1)
59.  LL=INT(M-I)
60.  IF (J.EQ.LL) GO TO 21
61.  19 B=Y(J)
62.  Y(J)=Y(LL)
63.  Y(LL)=B
64.  21 CONTINUE
65.  C UNNORMALIZED EIGENVECTOR OF ORIGINAL MATRIX OBTAINED
66.  B=0.0
67.  DO22 I=1,N
68.  IF(ABS(B).GE. ABS(Y(I)))GO TO 22
69.  23 B=Y(I)
70.  22 CONTINUE
71.  DO24 I=1,N

```

RVEC - 1

29 MAR 76 G.O.-91


```

72.      24 Y(I)=Y(I)/B
73.      C   NORMALIZED EIGENVECTOR OBTAINED
74.      C PUT Y(I) (REAL EIGENVECTOR) IN EIGVECT(NUMOFEV,I)
75.      DO 100 I=1,N
76.      100 EIGVEC(NUM,I)=Y(I)
77.      RETURN
78.      END

```

29 MAR 76 G.04-01

RVEC-2

ORIGINAL PAGE IS
OF POOR QUALITY

8J

IMVEC

```

1.  SUBROUTINE IMVEC(A,NMAX,INT,C1,AMU,SMALL,N,NUM,EIGVEC,NVEC,INTER,
2.  IV,Z,B)
3.  DIMENSION A(NMAX,NMAX),INTER(1),V(1),Z(1),INT(1),B(NMAX,NMAX),
4.  IEIGVEC(NMAX,NMAX)
5.  DO50 I=1,N
6.  DO50 J=1,N
7.  50 B(I,J)=0.0
8.  DO1 I=1,N
9.  A(I,I)=A(I,I)-C1
10. 1 B(I,I)=B(I,I)-AMU
11.  DO2 I=2,N
12.  C=A(I-1,I-1)+A(I-1,I-1)*B(I-1,I-1)+B(I-1,I-1)
13.  D=A(I,I-1)+A(I,I-1)*B(I-1,I-1)+B(I,I-1)
14.  IF (C.GE.0) GO TO 4
15.  3 INTER(I)=I
16.  G=C
17.  C=D
18.  D=G
19.  J=I-1
20.  DO5 K=J,N
21.  G=A(I-1,K)
22.  A(I-1,K)=A(I,K)
23.  A(I,K)=G
24.  G=B(I-1,K)
25.  B(I-1,K)=B(I,K)
26.  5 B(I,K)=G
27.  GO TO 6
28.  4 INTER(I)=I-1
29.  6 IF (D.EQ.0.) GO TO 2
30.  7 G=A(I,I-1)
31.  A(I,I-1)=-(G*A(I-1,I-1)+B(I,I-1)+B(I-1,I-1))/C
32.  B(I,I-1)=-(B(I,I-1)+A(I-1,I-1)+G*B(I-1,I-1))/C
33.  DO8 K=I,N
34.  A(I,K)=A(I,K)+A(I,I-1)*A(I-1,K)-B(I,I-1)*B(I-1,K)
35.  8 B(I,K)=B(I,K)+B(I,I-1)*A(I-1,K)+A(I,I-1)*B(I-1,K)
36.  2 CONTINUE
37.  C TRIANGULAR MATRIX FORMED
38.  DO9 I=1,N
39.  C=A(I,I)+A(I,I)*B(I,I)+B(I,I)
40.  IF (C.GE.SMALL) GO TO 9
41.  A(I,I)=SQRT(SMALL)
42.  B(I,I)=0.0
43.  9 CONTINUE
44.  DO 1111 I=1,N
45.  Y(I)=1.0
46.  1111 Z(I)=0.0
47.  DO 8888 J=1,N
48.  C=A(N,N)+A(N,N)*B(N,N)+B(N,N)
49.  G=Y(N)
50.  Y(N)=(G*A(N,N)+Z(N)+B(N,N))/C
51.  Z(N)=(Z(N)+A(N,N)-G*B(N,N))/C
52.  DO17 I=2,N
53.  E=0.0
54.  F=0.0
55.  M=N+1-I
56.  DO18 J=2,I
57.  L=N+J-I
58.  E=E+Y(L)*A(M,L)-Z(L)*B(M,L)
59.  18 F=F+Y(L)*B(M,L)+Z(L)*A(M,L)
60.  D=A(M,M)+A(M,M)*B(M,M)+B(M,M)
61.  G=Y(M)
62.  Y(M)=((G-E)*A(M,M)+(Z(M)-F)*B(M,M))/D
63.  17 Z(M)=((Z(M)-F)*A(M,M)-(G-E)*B(M,M))/D
64.  E=L.0
65.  DO51 I=1,N
66.  F=Y(I)*Y(I)+Z(I)*Z(I)
67.  IF (E.GE.F) GO TO 51
68.  52 E=F
69.  L=I
70.  51 CONTINUE
71.  A1=Y(L)
72.  A2=Z(L)
73.  DO14 I=1,N

```

IMVEC-1

29 MAR 76 0.04-01

ORIGINAL PAGE IS
OF POOR QUALITY

9J

```

74.      G=Y(I)
75.      Y(I)=(G*A1+Z(I)*A2)/E
76.      14 Z(I)=(Z(I)*A1-G*A2)/E
77.      IF(JJJ.EQ.4) GO TO 6666
78.      DO15 I=2,N
79.      IF (INTER(I).NE.1) GO TO 30
80.      16 G=Y(I)
81.      Y(I)=Y(I-1)
82.      Y(I-1)=G
83.      G=Z(I)
84.      Z(I)=Z(I-1)
85.      Z(I-1)=G

86.      30 Y(I)=Y(I)+A(I,I-1)*Y(I-1)-B(I,I-1)*Z(I-1)
87.      15 Z(I)=Z(I)+B(I,I-1)*Y(I-1)+A(I,I-1)*Z(I-1)
88.      8888 CONTINUE
89.      C      EIGENVECTOR OF HESSENBERG MATRIX OBTAINED

90.      6666 M=N-1
91.      DO21 I=2,M
92.      J=N+1-I
93.      L=J+1
94.      DO20 K=L,N
95.      Y(K)=Y(K)-Y(J)*A(K,J-1)+Z(J)*B(K,J-1)
96.      20 Z(K)=Z(K)-Z(J)*A(K,J-1)+Y(J)*B(K,J-1)
97.      LL=INT(N-I)
98.      IF (J.EQ.LL) GO TO 21
99.      19 G=Y(J)
100.     Y(J)=Y(LL)
101.     Y(LL)=G
102.     G=Z(J)
103.     Z(J)=Z(LL)
104.     Z(LL)=G

105.     21 CONTINUE
106.     C      UNNORMALIZED EIGENVECTOR OBTAINED
107.     E=0.0
108.     DO22 I=1,N
109.     F=Y(I)*Y(I)+Z(I)*Z(I)
110.     IF (E.GE.F) GO TO 22
111.     23 E=F
112.     L=I

113.     22 CONTINUE
114.     A1=Y(L)
115.     A2=Z(L)
116.     DO24 I=1,N
117.     G=Y(I)
118.     Y(I)=(G*A1+Z(I)*A2)/E
119.     Z(I)=(Z(I)*A1-G*A2)/E
120.     C      NORMALIZED EIGENVECTOR OBTAINED
121.     C PUT Y(I) (REAL PART) IN EIGVECT(NUMOFEV,I)
122.     DO 100 I=1,N
123.     EIGVEC(NUM,I)=Y(I)
124.     NUM=NUM+1
125.     C PUT Z(I) (COMPLEX PART) IN EIGVECT(NUMOFEV,I)
126.     DO 101 I=1,N
127.     EIGVEC(NUM,I)=Z(I)
128.     RETURN
129.     END

```

6666

30

21

22

IMVEC-2

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	U	TYPE	VAR DIM
A		Argument 1 to subprogram CALCEV.	CALCEV	M	REAL A	(NMAX,NMAX
A		Argument 1 to subprogram COMPIT.	COMPIT	M	REAL A	(NMAX,NMAX
A		Argument 1 to subprogram IMVEC.	IMVEC	M	REAL A	(NMAX,NMAX
A		Local variable in subprogram MAIN.	MAIN	M	REAL A	(NMAX,NMAX
			CALCEV/1	M	REAL A	(NMAX,NMAX
			COMPIT/1	M	REAL A	(NMAX,NMAX
			IMVEC/1	M	REAL A	(NMAX,NMAX
			REALIT/1	M	REAL A	(NMAX,NMAX
			RVEC/1	M	REAL A	(NMAX,NMAX
A		Argument 1 to subprogram REALIT.	REALIT	M	REAL A	(NMAX,NMAX
A		Argument 1 to subprogram RVEC.	RVEC	M	REAL A	(NMAX,NMAX
AMU		Argument 5 to subprogram IMVEC.	IMVEC	I	REAL AMU	

29 MAR 76 G.09-01

A-1

ORIGINAL PAGE IS
OF POOR QUALITY

113

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	U	TYPE	VAR DIM
B		Argument 14 to subprogram IMVEC.	IMVEC	M	REAL B	(NMAX,NMAX)
BB		Local variable in subprogram MAIN.	MAIN IMVEC/14	1 M	REAL BB REAL B	(NMAX,NMAX)

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	DECLARATIONS			
			SUBR	U	TYPE	VAR
C		Argument 6 to subprogram REALIT.	REALIT	M	REAL	C (1)
CALCEV		Subprogram external name.	CALCEV	E		CALCEV
			MAIN	S		CALCEV
CC		Local variable in subprogram MAIN.	MAIN	I	REAL	CC
			IMVEC/11	M	INTE	INTER (1)
			REALIT/6	M	REAL	C (1)
			RVEC/10	M	INTE	INTER (1)
COMPIT		Subprogram external name.	COMPIT	E		COMPIT
			MAIN	S		COMPIT
C1		Argument 4 to subprogram INVEC.	INVEC	I	REAL	C1
C1		Local variable in subprogram MAIN.	MAIN	M	REAL	C1
			IMVEC/4	I	REAL	C1
			RVEC/4	I	REAL	C1
C1		Argument 4 to subprogram RVEC.	RVEC	I	REAL	C1
C2		Local variable in subprogram MAIN.	MAIN	M	REAL	C2
			IMVEC/5	I	REAL	AMU

29 MAR 76 G.09-01

C-1

ORIGINAL PAGE IS
OF POOR QUALITY

13J

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE				USAGE	
			SUBR	U	TYPE	VAR	DIM	
DEL		Argument 3 to subprogram CALCEV.						CALCEV 1 REAL DEL
DEL		Local variable in subprogram MAIN.						MAIN 4 REAL DEL CALCEV/3 1 REAL DEL

29 MAR 76 G.04-01

D-1

14J

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	U	TYPE	VAR DIM
EIGVEC		Argument 9 to subprogram IMVEC.	IMVEC	0	REAL	EIGVEC(NMAX,NMAX)
EIGVEC		Local variable in subprogram MAIN.	MAIN	1	REAL	EIGVEC(NMAX,NMAX)
			IMVEC/9	0	REAL	EIGVEC(NMAX,NMAX)
			RVEC/8	0	REAL	EIGVEC(NMAX,NMAX)
EIGVEC		Argument 8 to subprogram RVEC.	RVEC	0	REAL	EIGVEC(NMAX,NMAX)

29 MAR 76 G.09-01

E-1

ORIGINAL PAGE IS
OF POOR QUALITY

15J

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	U	TYPE	VAR DIR
IJK		Argument 5 to subprogram CALCEV.	CALCEV M INTE IJK			
IJK		Local variable in subprogram MAIN.	MAIN CALCEV/5	W M	INTE INTE	IJK IJK
IMVEC		Subprogram external name.	IMVEC MAIN	E S		IMVEC IMVEC
INT		Argument 3 to subprogram IMVEC.	IMVEC I INTE INT (1)			
INT		Local variable in subprogram MAIN.	MAIN IMVEC/3 RVEC/3	W I I	INTE INTE INTE	INT (1) (1) (1)
INT		Argument 3 to subprogram RVEC.	RVEC I INTE INT (1)			
INTER		Argument 11 to subprogram IMVEC.	IMVEC M INTE INTER (1)			
INTER		Argument 10 to subprogram RVEC.	RVEC M INTE INTER (1)			

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	U	TYPE	VAR DIM
L		Argument 3 to subprogram COMPIT.				COMPIT I INTE L
L		Local variable in subprogram MAIN.				MAIN W INTE L COMPIT/3 I INTE L REALIT/3 I INTE L
L		Argument 3 to subprogram REALIT.				REALIT I INTE L

29 MAR 76 G.04-01

L-1

ORIGINAL PAGE IS
OF POOR QUALITY

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SOAK	U TYPE	VAR	DTM
M		Argument 7 to subprogram CALCEV.				CALCEV 1 INTE M

29 MAR 76 G.04-01

M-1

18J

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	O	TYPE	VAR DIM
N		Argument 6 to subprogram CALCEV.	CALCEV	I	INTE	N
N		Argument 4 to subprogram COMPIT.	COMPIT	I	INTE	N
N		Argument 7 to subprogram IMVEC.	IMVEC	I	INTE	N
N		Local variable in subprogram MAIN.	MAIN	M	INTE	N
			CALCEV/6	I	INTE	N
			COMPIT/4	I	INTE	N
			REALIT/4	I	INTE	N
N		Argument 4 to subprogram REALIT.	REALIT	I	INTE	N
N		Argument 6 to subprogram RVEC.	RVEC	I	INTE	N
NMAX		Argument 2 to subprogram CALCEV.	CALCEV	I	INTE	NMAX
NMAX		Argument 2 to subprogram COMPIT.	COMPIT	I	INTE	NMAX
NMAX		Argument 2 to subprogram IMVEC.	IMVEC	I	INTE	NMAX
NMAX		Local variable in subprogram MAIN.	MAIN	I	INTE	NMAX
			CALCEV/2	I	INTE	NMAX
			COMPIT/2	I	INTE	NMAX
			IMVEC/2	I	INTE	NMAX
			REALIT/2	I	INTE	NMAX
			RVEC/2	I	INTE	NMAX
NMAX		Argument 2 to subprogram REALIT.	REALIT	I	INTE	NMAX
NMAX		Argument 2 to subprogram RVEC.	RVEC	I	INTE	NMAX
NN		Local variable in subprogram MAIN.	MAIN	I	INTE	NN
			IMVEC/7	I	INTE	N
			RVEC/6	I	INTE	N
NUM		Argument 8 to subprogram IMVEC.	IMVEC	M	INTE	SUM
NUM		Local variable in subprogram MAIN.	MAIN	M	INTE	NUM
			IMVEC/8	M	INTE	NUM
			RVEC/7	I	INTE	NUM
NUM		Argument 7 to subprogram RVEC.	RVEC	I	INTE	NUM

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	U	TYPE	VAR
REALIT		Subprogram external name.	MAIN	S		REALIT
			REALIT	E		REALIT
RHO		Argument 5 to subprogram COMPIT.	COMPIT	I	REAL	RHO
RHO		Local variable in subprogram MAIN.	MAIN	W	REAL	RHO
			COMPIT/5	I	REAL	RHO
RVEC		Subprogram external name.	MAIN	S		RVEC
			RVEC	E		RVEC

29 MAR 76 G.04-01

R-1

20J

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	U	TYPE	VAR DIM
S		Argument 7 to subprogram REALIT.	REALIT	M	REAL	S (1)
SHIFT		Local variable in subprogram MAIN.	MAIN REALIT/5	W I	REAL REAL	SHIFT SHIFT
SHIFT		Argument 5 to subprogram REALIT.	REALIT	I	REAL	SHIFT
SIGMA		Argument 6 to subprogram COMPIT.	COMPIT	I	REAL	SIGMA
SIGMA		Local variable in subprogram MAIN.	MAIN COMPIT/6	W I	REAL REAL	SIGMA SIGMA
SMALL		Argument 6 to subprogram IMVEC.	IMVEC	I	REAL	SMALL
SMALL		Local variable in subprogram MAIN.	MAIN IMVEC/6 RVEC/5	W I I	REAL REAL REAL	SMALL SMALL SMALL
SMALL		Argument 5 to subprogram RVEC.	RVEC	I	REAL	SMALL
SQRT		<i>LIBRARY/SYSTEM</i> non-user-supplied routine.	CALCEV COMPIT IMVEC MAIN REALIT	F F F F F		SQRT SQRT SQRT SQRT SQRT
SS		Local variable in subprogram MAIN.	MAIN IMVEC/12 REALIT/7 RVEC/11	I M M M	REAL REAL REAL REAL	SS Y (1) S (1) Y (1)

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	U	TYPE	VAR DIM
VEC		Argument 4 to subprogram CALCEV.	CALCEV	0	REAL	VEC (1)
VEC		Local variable in subprogram MAIN.	MAIN CALCEV/4	1 0	REAL REAL	VEC VEC (1) (1)

29 MAR 76 G.04-01

V-1

22J

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	ROUTINE				USAGE	
			SUBR	U	TYPE	VAR	DIM	
y		Argument 12 to subprogram IMVEC.	IMVEC	M	REAL	Y	(1)	
y		Argument 11 to subprogram RVEC.	RVEC	M	REAL	Y	(1)	

29 MAR 76 G.04-01

ORIGINAL PAGE IS
OF POOR QUALITY

Y-1

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION	SUBROUTINE USAGE			
			SUBR	U TYPE	VAR	DIM
Z		Argument 13 to subprogram IMVEC.	IMVEC	M	REAL Z	(1)
Z		Local variable in subprogram MAIN.	MAIN IMVEC/13	I M	REAL Z REAL Z	(1)

29 MAR 76 G.04-01

Z-1

ORIGINAL PAGE IS
OF POOR QUALITY

24J

SUBROUTINE
MAIN

GLOSSARY FOR SUBROUTINE MAIN

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION
A		Working real local array, dimensioned (NMAX,NMAX). This symbol appears in the following calls - COMPIT argument # 1, ABS argument # 1, CALCEV argument # 1, RVEC argument # 1, IMVEC argument # 1, REALIT argument # 1.
AB		Working real local variable.
ANUM		Real local variable which is calculated but not used.
AR1		Working real local variable.
AR2		Working real local variable.
AR3		Working real local variable.
AR4		Working real local variable.
B		Working real local variable.
BB		Real local variable which is not explicitly calculated. This symbol appears in the following calls - IMVEC argument #14.
C		Working real local variable.
CALCEV		Called subroutine.
CC		Real local variable which is not explicitly calculated. This symbol appears in the following calls - RVEC argument #10, IMVEC argument #11, REALIT argument # 6.
COMPIT		Called subroutine.
CS		Working real local variable.
C1		Working real local variable. This symbol appears in the following calls - IMVEC argument # 4, RVEC argument # 4.
C2		Working real local variable. This symbol appears in the following calls - IMVEC argument # 5.
D		Working real local array, dimensioned (NMAX,NMAX).
DEL		Working real local variable. This symbol appears in the following calls - CALCEV argument # 3.
DELTA		Working real local variable.
DENOM		Working real local variable.
DISC		Working real local variable. This symbol appears in the following calls - ABS argument # 1, SQRT argument # 1.
E		Working real local variable.
EIGVEC		Real local array, dimensioned (NMAX,NMAX), which is not explicitly calculated. This symbol appears in the following calls - IMVEC argument # 9, RVEC argument # 8.
GREAT		Working real local variable.
I		Working integer local variable.
IJK		Working integer local variable. This symbol appears in the following calls - CALCEV argument # 5.
IMVEC		Called subroutine.
INT		Working integer local array, dimensioned (1). This symbol appears in the following calls - IMVEC argument # 3, RVEC argument # 2.
J		Working integer local variable.
JI		Working integer local variable.
K		Working integer local variable.

GLOSSARY FOR SUBROUTINE MAIN

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION
K1		Working Integer local variable.
K11		Working Integer local variable.
K1J		Working Integer local variable.
K2		Working Integer local variable.
L		Working Integer local variable. This symbol appears in the following calls - COMPIT argument # 3, REALIT argument # 3.
M		Working Integer local variable.
MAX		Working Integer local variable.
N		Working Integer local variable. This symbol appears in the following calls - REALIT argument # 4, CALCEV argument # 6, COMPIT argument # 4.
NM		Integer local variable which is calculated but not used.
NMAX		Integer local variable which is not explicitly calculated. This symbol appears in the following calls - IMVEC argument # 2, CALCEV argument # 2, COMPIT argument # 2, REALIT argument # 2, RVEC argument # 2.
NN		Integer local variable which is not explicitly calculated. This symbol appears in the following calls - IMVEC argument # 7, RVEC argument # 6.
NUM		Working Integer local variable. This symbol appears in the following calls - IMVEC argument # 8, RVEC argument # 7.
NVEC		Integer local variable which is not explicitly calculated. This symbol appears in the following calls - IMVEC argument #10, RVEC argument # 9.
N2I		Working Integer local variable.
R		Working real local variable.
REALIT		Called subroutine.
RMO		Working real local variable. This symbol appears in the following calls - COMPIT argument # 5.
RS		Working real local variable.
RVEC		Called subroutine.
R1		Working real local variable.
R2		Working real local variable.
SHIFT		Working real local variable. This symbol appears in the following calls - REALIT argument # 5.
SIGMA		Working real local variable. This symbol appears in the following calls - COMPIT argument # 6.
SMALL		Working real local variable. This symbol appears in the following calls - IMVEC argument # 6, RVEC argument # 5.
SQRT		Function reference.
SS		Real local variable which is not explicitly calculated. This symbol appears in the following calls - RVEC argument #11, IMVEC argument #12, REALIT argument # 7.
VEC		Real local array, dimensioned (1), which is not explicitly calculated. This symbol appears in the following calls - CALCEV argument # 4.

29 MAR 76 G.04-01

MAIN.2-2

ORIGINAL PAGE IS
OF POOR QUALITY

27J

GLOSSARY FOR SUBROUTINE MAIN

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION
Z		Real local variable which is not explicitly calculated. This symbol appears in the following calls - IMVEC argument #13.

29 MAR 76 0.04-01

MAIN.2-3

28J

SUBROUTINE
CALCEV

GLOSSARY FOR SUBROUTINE CALCEV

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION
A		<i>MODIFIED</i> real argument no. 1, dimensioned (NMAX,NMAX).
AI		Working real local variable.
B		Working real local variable.
C		Working real local variable.
CALCEV		Subprogram entry point.
CI		Working real local variable.
DEL		<i>INPUT</i> real argument no. 3.
DISC		Working real local variable. This symbol appears in the following calls - ABS argument # 1, SQRT argument # 1.
IJK		<i>MODIFIED</i> integer argument no. 5.
M		<i>INPUT</i> integer argument no. 7.
N		<i>INPUT</i> integer argument no. 6.
NMAX		<i>INPUT</i> integer argument no. 2.
R1		Working real local variable.
R2		Working real local variable.
SQRT		Function reference.
VEC		<i>OUTPUT</i> real argument no. 4, dimensioned (1).

29 MAR 76 G.04-01

ORIGINAL PAGE IS
OF POOR QUALITY

CALCEV.2-1

30J

SUBROUTINE
REALIT

GLOSSARY FOR SUBROUTINE REALIT

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION
A		<i>MODIFIED</i> real argument no. 1, dimensioned (NMAX,NMAX).
C		<i>MODIFIED</i> real argument no. 6, dimensioned (1).
DENOM		Working real local variable.
F		Working real local variable.
I		Working integer local variable.
II		Working integer local variable.
J		Working integer local variable.
L		<i>INPUT</i> integer argument no. 3.
M		Working integer local variable.
N		<i>INPUT</i> integer argument no. 4.
NMAX		<i>INPUT</i> integer argument no. 2.
REALIT		Subprogram entry point.
S		<i>MODIFIED</i> real argument no. 7, dimensioned (1).
SHIFT		<i>INPUT</i> real argument no. 5.
SQRT		Function reference.

SUBROUTINE
COMPIT

GLOSSARY FOR SUBROUTINE COMPIT

FORTHAN
SYMBOL

MAIN
SYMBOL

DESCRIPTION

A	<i>MODIFIED</i> real argument no. 1, dimensioned (NMAX,NMAX).
B1	Working real local variable.
B2	Working real local variable.
B3	Working real local variable.
COMPIT	Subprogram entry point.
C1	Working real local variable.
C2	Working real local variable.
D	Working real local variable.
E	Working real local variable.
G	Working real local variable.
I	Working integer local variable.
J	Working integer local variable.
K	Working integer local variable.
L	<i>INPUT</i> Integer argument no. 3.
M	Working integer local variable.
N	<i>INPUT</i> Integer argument no. 4.
NMAX	<i>INPUT</i> Integer argument no. 2.
RHO	<i>INPUT</i> real argument no. 5.
SIGMA	<i>INPUT</i> real argument no. 6.
SQRT	Function reference.

SUBROUTINE
RVEC

GLOSSARY FOR SUBROUTINE RVEC

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION
A		<i>MODIFIED</i> real argument no. 1, dimensioned (NMAX,NMAX). This symbol appears in the following calls - ABS argument # 1.
B		Working real local variable. This symbol appears in the following calls - ABS argument # 1.
C		Working real local variable.
CI		<i>INPUT</i> real argument no. 4.
E		Working real local variable.
EIGVEC		<i>OUTPUT</i> real argument no. 8, dimensioned (NMAX,NMAX).
I		Working integer local variable.
INT		<i>INPUT</i> integer argument no. 3, dimensioned (1).
INTER		<i>MODIFIED</i> integer argument no.10, dimensioned (1).
J		Working integer local variable.
JJJJ		Working integer local variable.
K		Working integer local variable.
L		Working integer local variable.
LL		Working integer local variable.
M		Working integer local variable.
N		<i>INPUT</i> integer argument no. 6.
NMAX		<i>INPUT</i> integer argument no. 2.
NUM		<i>INPUT</i> integer argument no. 7.
RVEC		Subprogram entry point.
SMALL		<i>INPUT</i> real argument no. 5.
Y		<i>MODIFIED</i> real argument no.11, dimensioned (1). This symbol appears in the following calls - ABS argument # 1.

29 MAR 76 6.04-01

RVEC.2-1

ORIGINAL PAGE IS
OF POOR QUALITY

35J

SUBROUTINE
IMVEC

GLOSSARY FOR SUBROUTINE IMVEC

FORTRAN SYMBOL	MATH SYMBOL	DESCRIPTION
A		<i>MODIFIED</i> real argument no. 1, dimensioned (NMAX,NMAX).
AMU		<i>INPUT</i> real argument no. 5.
A1		Working real local variable.
A2		Working real local variable.
B		<i>MODIFIED</i> real argument no.14, dimensioned (NMAX,NMAX).
C		Working real local variable.
C1		<i>INPUT</i> real argument no. 4.
D		Working real local variable.
E		Working real local variable.
EIGVEC		<i>OUTPUT</i> real argument no. 9, dimensioned (NMAX,NMAX).
F		Working real local variable.
G		Working real local variable.
I		Working integer local variable.
IMVEC		Subprogram entry point.
INT		<i>INPUT</i> integer argument no. 3, dimensioned (1).
INTER		<i>MODIFIED</i> integer argument no.11, dimensioned (1).
J		Working integer local variable.
JJJ		Working integer local variable.
K		Working integer local variable.
L		Working integer local variable.
LL		Working integer local variable.
M		Working integer local variable.
N		<i>INPUT</i> integer argument no. 7.
NMAX		<i>INPUT</i> integer argument no. 2.
NUM		<i>MODIFIED</i> integer argument no. 8.
SMALL		<i>INPUT</i> real argument no. 6. This symbol appears in the following calls - SQRT argument # 1.
SQRT		Function reference.
Y		<i>MODIFIED</i> real argument no.12, dimensioned (1).
Z		<i>MODIFIED</i> real argument no.13, dimensioned (1).

SUBROUTINE CALCEV

Inputs:

A DEL IJK M N NMAX

Outputs:

A IJK VEC

Subroutines Called:

SQRT

29 MAR 76 G.04-01

CALCEV.1-1

ORIGINAL PAGE IS
OF POOR QUALITY

39J

SUBROUTINE REALIT

Inputs:

A C L N NMAX S SHIFT

Outputs:

A C S

Subroutines Called:

SQRT

29 MAR 76 G.04-01

REALIT.1-1

ORIGINAL PAGE IS
OF POOR QUALITY

40J

C.4

SUBROUTINE REALIT

Inputs:

A C L N NMAX S SHIFT

Outputs:

A C S

Subroutines Called:

SQRT

29 MAR 76 G.04-01

REALIT.1-1

ORIGINAL PAGE IS
OF POOR QUALITY

40J

C.4

SUBROUTINE COMPIT

Inputs:

A L N NMAX RHO SIGMA

Outputs:

A

Subroutines Called:

SQRT

29 MAR 76 G.04-01

COMPIT.1-1

ORIGINAL PAGE IS
OF POOR QUALITY

41J

SUBROUTINE RVEC

Inputs:

A C1 INT INTER N NMAX NUM SMALL Y

Outputs:

A EIGVEC INTER Y

Subroutines Called:

29 MAR 76 8.04-01

RVEC.1-1

42J

SUBROUTINE IMVEC

Inputs:

A AMU B C1 INT INTER N NMAX NUM SMALL Y
Z

Outputs:

A B EIGVEC INTER NUM Y Z

Subroutines Called:

SQRT

Inputs:

Outputs:

Subroutines Called:

CALCEV COMPIT IMVEC REALIT RVEC SQRT

29 MAR 76 G.04-01

IMVEC.1-1

ORIGINAL PAGE IS
OF POOR QUALITY

43J

THE FOLLOWING IS A LIST OF THE USAGE CODES WHICH MAY APPEAR IN THE CROSS-REFERENCE MATRIX, EACH APPEARANCE OF A USAGE CODE INDICATES HOW THE SYMBOL TO THE LEFT IS USED IN THE SUBROUTINE LISTED ABOVE,

- A ARITHMETIC STATEMENT FUNCTION
- C CONSTANT - APPEARS IN DATA STATEMENT AND IS USED BUT VALUE IS NOT RE-CALCULATED.
- D DATA - APPEARS IN DATA STATEMENT BUT IS NOT USED.
- E SUBROUTINE ENTRY POINT.
- F FUNCTION REFERENCE.
- I FOR VARIABLE - INPUT, USED BUT NOT RE-CALCULATED,
FOR I/O FILE - USED IN READ STATEMENT.
- M FOR VARIABLE - MODIFIED, BOTH USED AND RE-CALCULATED,
APPLIES TO COMMON VARIABLES AND ARGUMENTS,
FOR I/O FILE - BOTH READ AND WRITTEN.
- O FOR VARIABLE - OUTPUT, VALUE IS CALCULATED BUT NOT USED,
FOR I/O FILE - USED IN ENDFILE, PRINT, FINCH, AND/OR
WRITE STATEMENT,
FOR FUNCTION NAME - FUNCTION ENTRY POINT (FUNCTION VALUE
IS OUTPUT).
- P I/O FILE POSITIONED - BACKSPACE OR REWIND.
- S SUBROUTINE CALL
- W WORKING VARIABLE - BOTH CALCULATED AND USED, APPLIES TO
LOCAL VARIABLES ONLY.
- (INTEGER) ARGUMENT NUMBER, THE VARIABLE AT LEFT APPEARS AS
THE I-TH ARGUMENT IN A CALL TO THE ABOVE SUBR.
- * ARGUMENT NUMBERS, TWO OR MORE INTEGERS
(ARGUMENT NUMBERS, SEE ABOVE) BELONG HERE
SO THE USER HAS TO CONSULT THE PROGRAM GLOSSARY
TO SEE WHAT THEY ARE.
- * THE COMMON BLOCK ON THE LEFT APPEARS IN THE SUBROUTINE
ABOVE, AT LEAST ONE ELEMENT OF THE COMMON BLOCK IS USED
IN SOME MANNER.

CROSS REFERENCE MATRIX

3/29/76 PAGE 2,01

SYMBOL	STORE	LOC	/	C	C	I	M	R	/	R
SHIFT	/MAIN	/(*)					W	5	
SIGMA	/MAIN	/(*)			6		W		
SMALL	/MAIN	/(*)				6	W		5
SS	/MAIN	/(*)				12	I	7	11
VEC	/MAIN	/(*)			4		I		
Z	/MAIN	/(*)				13	I		
A	/REALIT/(ARG	1)							M	
C	/REALIT/(ARG	6)							M	
L	/REALIT/(ARG	3)							I	
N	/REALIT/(ARG	4)							I	
NMAX	/REALIT/(ARG	2)							I	
REALIT	/REALIT/(\$)						S	E	
S	/REALIT/(ARG	7)							M	
SHIFT	/REALIT/(ARG	5)							I	
A	/RVEC	/(ARG	1)							M
C1	/RVEC	/(ARG	4)							I
EIGVEC	/RVEC	/(ARG	8)							0
INT	/RVEC	/(ARG	3)							I
INTER	/RVEC	/(ARG	10)							M
N	/RVEC	/(ARG	6)							I
NMAX	/RVEC	/(ARG	2)							I
NUM	/RVEC	/(ARG	7)							I
RVEC	/RVEC	/(S)					S		E
SMALL	/RVEC	/(ARG	5)							I
Y	/RVEC	/(ARG	11)							M
SQRT	/SORT	/(S)			F	F	F	F	F

CROSS REFERENCE MATRIX

3/29/76 PAGE 1,01

V T
E I C. I/
C P E N L/ C
L M V I A/ E
A O M A E/ V
C C I M R/ R

SYMBOL	STORE	LOC										
A	/CALCEV/(ARG 1)		M									
CALCEV	/CALCEV/(\$)		E	S								
DEL	/CALCEV/(ARG 3)		I									
IJK	/CALCEV/(ARG 5)		M									
M	/CALCEV/(ARG 7)		I									
N	/CALCEV/(ARG 6)		I									
NMAX	/CALCEV/(ARG 2)		I									
VEC	/CALCEV/(ARG 4)		O									
A	/COMPIT/(ARG 1)		I									
COMPIT	/COMPIT/(\$)		I	S								
L	/COMPIT/(ARG 3)		I									
N	/COMPIT/(ARG 4)		I									
NMAX	/COMPIT/(ARG 2)		I									
RHO	/COMPIT/(ARG 5)		I									
SIGMA	/COMPIT/(ARG 6)		I									
A	/IMVEC /(ARG 1)		M									
AMU	/IMVEC /(ARG 5)		I									
B	/IMVEC /(ARG14)		M									
C1	/IMVEC /(ARG 4)		I									
EIGVEC	/IMVEC /(ARG 9)		O									
IMVEC	/IMVEC /(\$)		E	S								
INT	/IMVEC /(ARG 3)		I									
INTER	/IMVEC /(ARG11)		M									
N	/IMVEC /(ARG 7)		I									
NMAX	/IMVEC /(ARG 2)		I									
NUM	/IMVEC /(ARG 8)		M									
SMALL	/IMVEC /(ARG 6)		I									
Y	/IMVEC /(ARG12)		M									
Z	/IMVEC /(ARG13)		M									
A	/MAIN /(*)		1	1	1	W	1		1			
BB	/MAIN /(*)		14	I								
CC	/MAIN /(*)		11	I	6		10					
C1	/MAIN /(*)		4	W			4					
C2	/MAIN /(*)		5	W								
DEL	/MAIN /(*)		3	W								
EIGVEC	/MAIN /(*)		9	I			8					
IJK	/MAIN /(*)		5	W								
INT	/MAIN /(*)		3	W			3					
L	/MAIN /(*)		3	W	3							
N	/MAIN /(*)		6	4	W	4						
NMAX	/MAIN /(*)		2	2	2	I	2		2			
NN	/MAIN /(*)		7	I					6			
NUM	/MAIN /(*)		8	W					7			
NVEC	/MAIN /(*)		10	I					9			
RHO	/MAIN /(*)		5	W								

SOURCE PROGRAM LISTING

\$ IN THE LABEL FIELD INDICATES THE EXPANSION
OF THE LOGICAL IF STATEMENT ON THE PREVIOUS LINE

LINE	STMT	ALOCK	SOURCE
1	1	1	SUBROUTINE UNSEIG(A,NMAX,NN,NVEC,VEC,EIGVEC,INT,CC,SS,Z,D,BB)
2	0	0	C FOR DESCRIPTION OF METHOD USED IN THIS PROGRAM, SEE **THE QR
3	0	0	C TRANSFORMATION** BY FRANCIS, COMPUTER JOURNAL, VOL. 4 PP. 265,332
4	0	0	C START HERE TO PRODUCE UPPER HESSENBURG MATRIX
5	2	1	DIMENSION A(NMAX,NMAX),INT(1),VEC(1),D(NMAX,NMAX),EIGVEC(NMAX,NMAX
6	2	1	C)
7	3	2	N=NN
8	4	3	K=N-2
9	5	4	DO1J=1,K
10	6	5	K1=J+1
11	7	6	K2=J+2
12	8	7	GREAT= ABS(A(K1,J))
13	9	8	MAX=K1
14	10	9	DO2I=K2,N
15	11	10	AB= ABS(A(I,J))
16	12	11	IF (GREAT.GE.AB)
16	13	12	\$ GO TO 2
17	14	13	3 GREAT=AB
18	15	14	MAX=I
19	16	15	2 CONTINUE
20	0	0	C INT(I) GIVES INFORMATION ABOUT INTERCHANGES
21	17	16	INT(J)=MAX
22	18	17	IF (MAX.EQ.K1)
22	19	18	\$ GO TO 5
23	20	19	4 DO6L=J,N
24	21	20	E=A(K1,L)
25	22	21	A(K1,L)=A(MAX,L)
26	23	22	6 A(MAX,L)=E
27	24	23	DO7L=1,N
28	25	24	E=A(L,K1)
29	26	25	A(L,K1)=A(L,MAX)
30	27	26	7 A(L,MAX)=E
31	0	0	C INTERCHANGE DONE
32	28	27	5 DO8I=K2,N
33	29	28	IF (A(I,J).EQ.0.)
33	30	29	\$ GO TO 8
34	31	30	A(I,J)=-A(I,J)/A(K1,J)
35	32	31	DO11M=K1,N
36	33	32	11 A(I,M)=A(I,M)+A(I,J)*A(K1,M)
37	34	33	8 CONTINUE
38	0	0	C LEFT MULT. DONE
39	35	34	DO 2000 M=K2,N
40	36	35	IF (A(M,J).EQ.0.)
40	37	36	\$ GO TO 2000
41	38	37	10 DO 2001 I=1,N
42	39	38	2001 A(I,K1)=A(I,K1)-A(I,M)*A(M,J)
43	40	39	2000 CONTINUE
44	41	40	1 CONTINUE
45	0	0	C RIGHT MULT. DONE

ORIGINAL PAGE IS
OF POOR QUALITY

```

46 0 0 C HESSENBURG MATRIX NOW FORMED AS A(I,J). MULTIPLIERS STORED WITH IT
47 42 41 DO 105 I=1,N
48 43 42 DO 105 J=1,N
49 44 43 105 D(I,J)=A(I,J)
50 0 0 C PUT APPROPRIATE ZEROS WHERE MULTIPLIERS WERE STORED
51 45 44 DO 106 I=3,N
52 46 45 L=I-2
53 47 46 DO 106 J=1,L
54 48 47 106 A(I,J)=0.0
55 49 48 K1=N+1
56 50 49 R=0.0
57 51 50 C=0.0
58 52 51 D033I=1,N
59 53 52 R=R+ ABS(A(1,I))
60 54 53 33 C=C+ ABS(A(1,N))
61 55 54 D034I=2,N
62 56 55 RS=0.0
63 57 56 CS=0.0
64 58 57 L=I-1
65 59 58 D035J=L,N
66 60 59 RS=RS+ ABS(A(I,J))
67 61 60 K1J=K1-J
68 62 61 K1I=K1-I
69 63 62 35 CS=CS+ ABS(A(K1J,K1I))
70 64 63 IF (RS.LE.R)
71 65 64 $ GO TO 36
72 66 65 37 R=RS
73 67 66 36 IF (CS.LE.C)
74 68 67 $ GO TO 34
75 69 68 44 C=CS
76 70 69 34 CONTINUE
77 71 70 IJK=1
78 72 71 IF (R.GE.C)
79 73 72 $ GO TO 54
80 0 0 C DELTA IS CRITERION FOR STOPPING AN ITERATION
81 74 73 53 DELTA=R*1.0E-11
82 75 74 C DEL IS ARBITRARY SHIFT
83 76 75 DEL=R/17.1829
84 77 76 GO TO 153
85 78 77 54 DELTA=C*1.0E-11
86 0 0 DEL=C/17.1829
87 0 0 C DELTA COMPUTED
88 79 78 C PRODUCE A-PI, ITERATE ON THIS TO BREAK UP EIGENVALUES OF EQUAL MODULUS
89 80 79 153 DO 107 I=1,N
90 81 80 107 A(I,I)=A(I,I)-DEL
91 82 81 55 IF (N.NE.1)
92 0 0 $ GO TO 48
93 0 0 C SUBROUTINE CALCEV CALCULATES EIGENVALUES OF 1X1 OR 2X2 MATRICES.
94 0 0 C PI IS ADDED BACK IN CALCEV
95 83 82 47 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
96 84 83 GO TO 76
97 85 84 48 IF (N.NE.2)
98 86 85 $ GO TO 80
99 87 86 49 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
100 88 87 GO TO 76
101 0 0 C QR TRANSFORMATION STARTS HERE
102 0 0 C INITIALIZE FOR DETERMINING SUBSEQUENT SHIFTS
103 89 88 80 AR1=0.0
104 90 89 AR3=0.0
105 91 90 AR4=0.0
106 0 0 C RE-ENTER AFTER REALIT OR COMPT SUBROUTINES CALLED
107 0 0 C SELECT FIRST (L2590) SUB-DIAGONAL ELEMENT FROM BOTTOM

```

```

103 92 91 61 L=N
104 93 92 0056 I=2,N
105 94 93 N2I=N+2-I
106 95 94 IF(ABS(A(N2I,N2I-1)).LT.DELTA)
106 96 95 $GO TO 57
107 97 96 56 L=L-1
108 0 0 C VALUE OF L SELECTED
109 98 97 57 IF (L.NE.N)
109 99 98 $ GO TO 59
110 100 99 58 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,1)
111 101 100 N=N-1
112 102 101 GO TO 48
113 103 102 59 IF ((L-N+1).NE.0)
113 104 103 $ GO TO 75
114 105 104 60 CALL CALCEV(A,NMAX,DEL,VEC,IJK,N,2)
115 106 105 N=N-2
116 107 106 GO TO 55
117 0 0 C DETERMINE EIGENVALUES OF 2X2 MATRIX -- ROW 1 = A(N-1,N-1) A(N-1,N)
118 0 0 C ROW 2 = A(N,N-1) A(N,N)
119 0 0 C AND BASE SHIFT ON THESE
120 108 107 75 B=A(N-1,N-1)*A(N,N)
121 109 108 C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
122 110 109 C1=A(N-1,N-1)-A(N,N)
123 111 110 DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)
124 112 111 IF (DISC.GE.0.)
124 113 112 $ GO TO 63
125 114 113 62 R1=0.5*B
126 115 114 R2=.5*SQRT(ABS(DISC))
127 116 115 ANUM=(R1-AR3)*(R1-AR3)+(R2-AR4)*(R2-AR4)
128 117 116 DENOM=AR3*AR3+AR4*AR4
129 118 117 IF (DENOM.NE.0.)
129 119 118 $ GO TO 200
130 120 119 201 RHO=0.0
131 121 120 SIGMA=0.0
132 122 121 GO TO 203
133 123 122 200 RHO=C
134 124 123 SIGMA=B
135 125 124 203 AR3=R1
136 126 125 AR4=R2
137 127 126 CALL COMPT(A,NMAX,L,N,RHO,SIGMA)
138 128 127 GO TO 61
139 129 128 63 IF (B.GE.0.)
139 130 129 $ GO TO 65
140 131 130 64 R1=.5*(B-SQRT(DISC))
141 132 131 GO TO 66
142 133 132 65 R1=.5*(B+SQRT(DISC))
143 134 133 IF (R1.NE.0.)
143 135 134 $ GO TO 66
144 136 135 83 R2=0.0
145 137 136 GO TO 84
146 138 137 66 R2=C/R1
147 0 0 C ROOTS OF 2*2 MATRIX FOUND
148 139 138 84 IF (ABS(A(N,N)-R1).GE.ABS(A(N,N)-R2))
148 140 139 $GO TO 68
149 141 140 67 AR2=R1
150 142 141 GO TO 69
151 143 142 68 AR2=R2
152 144 143 69 IF (AR1.NE.0.)
152 145 144 $ GO TO 71
153 146 145 72 SHIFT=0.0
154 147 146 GO TO 73
155 148 147 71 SHIFT=AR2

```

```

156 149 148      73 AR1=AR2
157 151 149      CALL REALIT(A,NMAX,L,N,SHIFT,CC,SS)
158 151 150      GO TO 61
159 0 0 0 C ARE VECTORS NEEDED
160 152 151      76 IF (NVEC.EQ.0)
160 153 152      $ GO TO 600
161 0 0 0 C INVERSE ITERATION IS USED FOR EIGENVECTORS
162 154 153      550 SMALL=DELTA*1.0E-30
163 155 154      NUM=0
164 156 155      JI=1
165 157 156      NM=NN+NN
166 0 0 0 C C1 IS REAL PART OF EIGENVALUE
167 158 157      513 C1=VEC(JI)
168 0 0 0 C C2 IS COMPLEX PART OF EIGENVALUE
169 159 158      C2=VEC(JI+1)
170 160 159      NUM=NUM+1
171 0 0 0 C REPLACE HESSENBURG MATRIX AND MULTIPLIERS
172 161 160      DO 570 I=1,NN
173 162 161      DO 570 J=1,NN
174 163 162      570 A(I,J)=D(I,J)
175 164 163      IF (C2.NE.0.)
175 165 164      $ GO TO 561
176 166 165      CALL RVEC(A,NMAX,INT,C1,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS)
177 167 166      JI=JI+2
178 168 167      IF (NVEC.EQ.NUM)
178 169 168      $GO TO 600
179 170 169      GO TO 513
180 171 170      561 CALL IMVEC(A,NMAX,INT,C1,C2,SMALL,NN,NUM,EIGVEC,NVEC,CC,SS,Z,BB)
181 0 0 0 C NEXT STATEMENT IGNORES CONJUGATE COMPLEX EIGENVALUES
182 172 171      JI=JI+4
183 173 172      IF (NVEC.EQ.NUM)
183 174 173      $GO TO 600
184 175 174      GO TO 513
185 176 175      600 RETURN
186 177 1 1      END

```


\$ IN THE LABEL FIELD INDICATES THE EXPANSION
OF THE LOGICAL IF STATEMENT ON THE PREVIOUS LINE

LINE	STMT	ALOCK	SOURCE
187	1	1	SUBROUTINE CALCEV(A,NMAX,DEL,VEC,IJK,N,M)
188	2	1	DIMENSION A(NMAX,NMAX),VEC(1)
189	3	2	IF (M.NE.1)
189	4	3	\$ GO TO 2
190	5	4	1 AI=0.0
191	6	5	A(N,N)=A(N,N)+DEL
192	0	0	C STORE EIGENVALUES IN VEC
193	7	6	VEC(IJK)=A(N,N)
194	8	7	VEC(IJK+1)=AI
195	9	8	IJK=IJK+2
196	10	9	RETURN
197	11	10	2 B=A(N-1,N-1)+A(N,N)
198	12	11	C=A(N-1,N-1)*A(N,N)-A(N,N-1)*A(N-1,N)
199	13	12	C1=A(N-1,N-1)-A(N,N)
200	14	13	DISC=C1*C1+4.0*A(N,N-1)*A(N-1,N)
201	15	14	IF (DISC.LT.0.)
201	16	15	\$ GO TO 9
202	17	16	4 AI=0.0
203	18	17	IF (B.GE.0.)
203	19	18	\$ GO TO 6
204	20	19	R1=0.5*(B- SQRT(DISC))
205	21	20	GO TO 7
206	22	21	6 R1=0.5*(B+ SQRT(DISC))
207	23	22	IF (R1.NE.0.)
207	24	23	\$ GO TO 7
208	25	24	10 R2=0.0
209	26	25	GO TO 11
210	27	26	7 R2=C/R1
211	28	27	11 R1=R1+DEL
212	29	28	R2=R2+DEL
213	30	29	VEC(IJK)=R2
214	31	30	VEC(IJK+1)=AI
215	32	31	VEC(IJK+2)=R1
216	33	32	VEC(IJK+3)=AI
217	34	33	GO TO 8
218	35	34	9 R1=0.5*B
219	36	35	AI=-.5* SQRT(ABS(DISC))
220	37	36	R1=R1+DEL
221	38	37	VEC(IJK)=R1
222	39	38	VEC(IJK+1)=AI
223	40	39	VEC(IJK+2)=R1
224	41	40	VEC(IJK+3)=AI
225	42	41	8 IJK=IJK+4
226	43	42	RETURN
227	44	1	END

ORIGINAL PAGE IS
OF POOR QUALITY

SOURCE PROGRAM LISTING

\$ IN THE LABEL FIELD INDICATES THE EXPANSION
OF THE LOGICAL IF STATEMENT ON THE PREVIOUS LINE

LINE	STMT	BLOCK	SOURCE
228	1	1	SUBROUTINE REALIT(A,NMAX,L,N,SHIFT,C,S)
229	2	1	DIMENSION A(NMAX,NMAX),C(1),S(1)
230	3	2	IF (SHIFT.EQ.0.)
230	4	3	\$ GO TO 2
231	5	4	1 DO3 I=L,N
232	6	5	3 A(I,I)=A(I,I)-SHIFT
233	7	6	2 M=N-1
234	8	7	DO4 I=L,M
235	9	8	DENOM= SQRT(A(I,I)**2+A(I+1,I)**2)
236	10	9	IF (DENOM.GT.0.)
236	11	10	\$ GO TO 10
237	12	11	11 C(I)=1.0
238	13	12	S(I)=0.0
239	14	13	GO TO 4
240	15	14	10 C(I)=A(I,I)/DENOM
241	16	15	S(I)=A(I+1,I)/DENOM
242	17	16	A(I,I)=DENOM
243	18	17	12 I=I+1
244	19	18	DO 35 J=I+1,N
245	20	19	F=A(I,J)
246	21	20	A(I,J)=A(I+1,J)*S(I)+F*C(I)
247	22	21	A(I+1,J)=A(I+1,J)*C(I)-F*S(I)
248	23	22	35 CONTINUE
249	24	23	4 CONTINUE
250	0	0	C TRIANGULAR MATRIX FORMED
251	25	24	DO5 J=L,M
252	26	25	DO6 I=L,J
253	27	26	F=A(I,J)
254	28	27	A(I,J)=A(I,J+1)*S(J)+F*C(J)
255	29	28	6 A(I,J+1)=A(I,J+1)*C(J)-F*S(J)
256	30	29	A(J+1,J)=A(J+1,J+1)*S(J)
257	31	30	5 A(J+1,J+1)=A(J+1,J+1)*C(J)
258	0	0	C TRANSFORMATION COMPLETED
259	32	31	IF (SHIFT.EQ.0.)
259	33	32	\$ RETURN
260	34	33	7 DO9 I=L,N
261	35	34	9 A(I,I)=A(I,I)+SHIFT
262	36	35	8 RETURN
263	37	1	END

SOURCE PROGRAM LISTING

\$ IN THE LABEL FIELD INDICATES THE EXPANSION
OF THE LOGICAL IF STATEMENT ON THE PREVIOUS LINE

LINE	STMT	ALOCK	SOURCE
264	1	1	SUBROUTINE COMPT(A,NMAX,L,N,RHO,SIGMA)
265	2	1	DIMENSION A(NMAX,NMAX)
266	0	0	C HOUSEHOLDER TYPE MATRICES USED INSTEAD OF ROTATIONS(SEE FRANCIS PAPER)
267	3	2	M=N-1
268	4	3	DO 1 I=L,M
269	5	4	IF (I.NE.L)
269	6	5	\$ GO TO 3
270	7	6	2 B1=A(L,L)*(A(L,L)-SIGMA)+A(L,L+1)*A(L+1,L)+RHO
271	8	7	B2=A(L+1,L)*(A(L,L)+A(L+1,L+1)-SIGMA)
272	9	8	B3=A(L+1,L)*A(L+2,L+1)
273	10	9	A(L+2,L)=0.0
274	11	10	GO TO 6
275	12	11	3 B1=A(I,I-1)
276	13	12	B2=A(I+1,I-1)
277	14	13	IF (I.EQ.M)
277	15	14	\$ GO TO 5
278	16	15	4 B3=A(I+2,I-1)
279	17	16	GO TO 6
280	18	17	5 B3=0.0
281	19	18	6 D= SQRT(B1*B1+B2*B2+B3*B3)
282	20	19	IF (B1.GE.0.)
282	21	20	\$ GO TO 8
283	22	21	7 D=-D
284	23	22	8 IF (D.EQ.0.)
284	24	23	\$ GO TO 1
285	25	24	9 C1=B2/(D+B1)
286	26	25	C2=B3/(D+B1)
287	27	26	E=2.0/(1.0+C1*C1+C2*C2)
288	28	27	DO 10 K=I,N
289	29	28	IF (I.EQ.M)
289	30	29	\$ GO TO 12
290	31	30	11 G=E*(A(I,K)+C1*A(I+1,K)+C2*A(I+2,K))
291	32	31	A(I,K)=A(I,K)-G
292	33	32	A(I+1,K)=A(I+1,K)-C1*G
293	34	33	A(I+2,K)=A(I+2,K)-C2*G
294	35	34	GO TO 10
295	36	35	12 G=E*(A(I,K)+C1*A(I+1,K))
296	37	36	A(I,K)=A(I,K)-G
297	38	37	A(I+1,K)=A(I+1,K)-C1*G
298	39	38	10 CONTINUE
299	40	39	IF (I.EQ.L)
299	41	40	\$ GO TO 14
300	42	41	13 A(I,I-1)=-D
301	43	42	A(I+1,I-1)=0.0
302	44	43	IF (I.EQ.M)
302	45	44	\$ GO TO 14
303	46	45	15 A(I+2,I-1)=0.0
304	0	0	C NOW OPERATION COMPLETED
305	47	46	14 J=I+2

ORIGINAL PAGE IS
OF POOR QUALITY

306	48	47	IF (J.LE.N)
306	49	48	5 GO TO 17
307	50	49	16 J=N
308	51	50	17 DO 18 K=L,J
309	52	51	IF (I.EQ.M)
309	53	52	5 GO TO 20
310	54	53	19 G=E*(A(K,I)+C1*A(K,I+1)+C2*A(K,I+2))
311	55	54	A(K,I)=A(K,I)-G
312	56	55	A(K,I+1)=A(K,I+1)-C1*G
313	57	56	A(K,I+2)=A(K,I+2)-C2*G
314	58	57	GO TO 18
315	59	58	20 G=E*(A(K,I)+C1*A(K,I+1))
316	60	59	A(K,I)=A(K,I)-G
317	61	60	A(K,I+1)=A(K,I+1)-C1*G
318	62	61	18 CONTINUE
319	63	62	J=I+3
320	64	63	IF (J.GT.N)
320	65	64	5 GO TO 1
321	66	65	21 G=E*C2*A(I+3,I+2)
322	67	66	A(I+3,I)=-G
323	68	67	A(I+3,I+1)=-C1*G
324	69	68	A(I+3,I+2)=A(I+3,I+2)-C2*G
325	0	0	C COLUMN OPERATION COMPLETED
326	70	69	1 CONTINUE
327	71	70	RETURN
328	72	1	END

SOURCE PROGRAM LISTING

5 IN THE LABEL FIELD INDICATES THE EXPANSION
OF THE LOGICAL IF STATEMENT ON THE PREVIOUS LINE

LINE	STMT	RLOCK	SOURCE
329	1	1	SUBROUTINE RVEC(A,NMAX,INT,C1,SMALL,N,NUM,EIGVEC,NVEC,INTER,Y)
330	2	1	DIMENSION A(NMAX,NMAX),INTER(1),Y(1),INT(1),EIGVEC(NMAX,NMAX)
331	3	2	DO1 I=1,N
332	4	3	1 A(I,I)=A(I,I)-C1
333	5	4	DO2 I=2,N
334	6	5	IF(ABS(A(I-1,I-1)).GE.ABS(A(I,I-1)))
334	7	6	\$GO TO 4
335	8	7	3 INTER(I)=I
336	9	8	J=I-1
337	10	9	DO5 K=J,N
338	11	10	B=A(I-1,K)
339	12	11	A(I-1,K)=A(I,K)
340	13	12	5 A(I,K)=B
341	14	13	GO TO 6
342	15	14	4 INTER(I)=I-1
343	16	15	6 IF (A(I,I-1).EQ.0.)
343	17	16	\$ GO TO 2
344	18	17	C=-A(I,I-1)/A(I-1,I-1)
345	19	18	A(I,I-1)=C
346	20	19	DO 8 K=I,N
347	21	20	8 A(I,K)=A(I,K)+C*A(I-1,K)
348	22	21	2 CONTINUE
349	0	0	C TRIANGULAR MATRIX FORMED
350	23	22	DO9 I=1,N
351	24	23	IF(ABS(A(I,I)).GE.SMALL)
351	25	24	\$GO TO 9
352	26	25	10 A(I,I)=SMALL
353	27	26	9 CONTINUE
354	0	0	C NORMALISED ((EIGENVECTOR)) OF HESSENBURG MATRIX OBTAINED FOR RHS = 1
355	28	27	DO 5555 I=1,N
356	29	28	5555 Y(I)=1.0
357	30	29	DO 8888 JJJJ=1,4
358	31	30	7777 Y(N)=Y(N)/A(N,N)
359	32	31	B=Y(N)
360	33	32	DO11 I=2,N
361	34	33	E=0.0
362	35	34	M=N+1-I
363	36	35	DO12 J=2,I
364	37	36	L=N+J-I
365	38	37	12 E=E+Y(L)*A(M,L)
366	39	38	Y(M)=(Y(M)-E)/A(M,M)
367	40	39	IF(ABS(B).GE.ABS(Y(M)))
367	41	40	\$GO TO 11
368	42	41	13 B=Y(M)
369	43	42	11 CONTINUE
370	44	43	DO14 I=1,N
371	45	44	14 Y(I)=Y(I)/B
372	46	45	IF(JJJJ.EQ.4)
372	47	46	\$GO TO 6666

```

373 48 47      DO15 I=2,N
374 49 48      IF (INTER(I).NE.I)
374 50 49      $ GO TO 15
375 51 50      16 B=Y(I)
376 52 51      Y(I)=Y(I-1)
377 53 52      Y(I-1)=B
378 54 53      15 Y(I)=Y(I)+A(I,I-1)*Y(I-1)
379 55 54      8888 CONTINUE
380 0 0      C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED (NOT NORMALIZED)
381 56 55      6666 M=N-1
382 57 56      DO21 I=2,M
383 58 57      J=N+1-I
384 59 58      L=J+1
385 60 59      DO20 K=L,N
386 61 60      20 Y(K)=Y(K)-Y(J)*A(K,J-1)
387 62 61      LL=INT(N-I)
388 63 62      IF (J.EQ.LL)
388 64 63      $ GO TO 21
389 65 64      19 B=Y(J)
390 66 65      Y(J)=Y(LL)
391 67 66      Y(LL)=B
392 68 67      21 CONTINUE
393 0 0      C UNNORMALIZED EIGENVECTOR OF ORIGINAL MATRIX OBTAINED
394 69 68      B=0.0
395 70 69      DO22 I=1,N
396 71 70      IF (ABS(B).GE. ABS(Y(I)))
396 72 71      $GO TO 22
397 73 72      23 B=Y(I)
398 74 73      22 CONTINUE
399 75 74      DO24 I=1,N
400 76 75      24 Y(I)=Y(I)/B
401 0 0      C      NORMALIZED EIGENVECTOR OBTAINED
402 0 0      C PUT Y(I) (REAL EIGENVECTOR) IN EIGVECT(NUMOFEV,I)
403 77 76      DO 100 I=1,N
404 78 77      100 EIGVEC(NUM,I)=Y(I)
405 79 78      RETURN
406 80 1      END

```

SOURCE PROGRAM LISTING

5 IN THE LABEL FIELD INDICATES THE EXPANSION
OF THE LOGICAL IF STATEMENT ON THE PREVIOUS LINE

LINE	STMT	BLOCK	SOURCE
407	1	1	SUBROUTINE IMVEC(A,NMAX,INT,C1,AMU,SMALL,N,NUM,EIGVEC,NVEC,INTER,
408	1	1	CY,Z,B)
409	2	1	DIMENSION A(NMAX,NMAX),INTER(1),Y(1),Z(1),INT(1),B(NMAX,NMAX),
410	2	1	CEIGVEC(NMAX,NMAX)
411	3	2	DO50 I=1,N
412	4	3	DO50 J=1,N
413	5	4	50 B(I,J)=0.0
414	6	5	DO1 I=1,N
415	7	6	A(I,I)=A(I,I)-C1
416	8	7	1 B(I,I)=B(I,I)-AMU
417	9	8	DO2 I=2,N
418	10	9	C=A(I-1,I-1)*A(I-1,I-1)+B(I-1,I-1)*B(I-1,I-1)
419	11	10	D=A(I,I-1)*A(I,I-1)+B(I,I-1)*B(I,I-1)
420	12	11	IF (C.GE.D)
420	13	12	5 GO TO 4
421	14	13	3 INTER(I)=I
422	15	14	G=C
423	16	15	C=D
424	17	16	D=G
425	18	17	J=I-1
426	19	18	DO5 K=J,N
427	20	19	G=A(I-1,K)
428	21	20	A(I-1,K)=A(I,K)
429	22	21	A(I,K)=G
430	23	22	G=B(I-1,K)
431	24	23	H(I-1,K)=B(I,K)
432	25	24	5 B(I,K)=G
433	26	25	GO TO 6
434	27	26	4 INTER(I)=I-1
435	28	27	6 IF (D.EQ.0.)
435	29	28	5 GO TO 2
436	30	29	7 G=A(I,I-1)
437	31	30	A(I,I-1)=-(G*A(I-1,I-1)+B(I,I-1)*B(I-1,I-1))/C
438	32	31	B(I,I-1)=-(B(I,I-1)*A(I-1,I-1)+G*B(I-1,I-1))/C
439	33	32	DO8 K=I,N
440	34	33	A(I,K)=A(I,K)+A(I,I-1)*A(I-1,K)-B(I,I-1)*B(I-1,K)
441	35	34	8 B(I,K)=B(I,K)+B(I,I-1)*A(I-1,K)+A(I,I-1)*B(I-1,K)
442	36	35	2 CONTINUE
443	0	0	C TRIANGULAR MATRIX FORMED
444	37	36	DO9 I=1,N
445	38	37	C=A(I,I)*A(I,I)+B(I,I)*B(I,I)
446	39	38	IF (C.GE.SMALL)
446	40	39	5 GO TO 9
447	41	40	A(I,I)= SQRT(SMALL)
448	42	41	d(I,I)=0.0
449	43	42	9 CONTINUE
450	44	43	DO 1111 I=1,N
451	45	44	Y(I)=1.0
452	46	45	1111 Z(I)=0.0

ORIGINAL PAGE IS
OF POOR QUALITY

```

453 47 46 DO 8888 JJJ=1,4
454 48 47 C=A(N,N)*A(N,N)+B(N,N)*B(N,N)
455 49 48 G=Y(N)
456 50 49 Y(N)=(G*A(N,N)+Z(N)*B(N,N))/C
457 51 50 Z(N)=(Z(N)*A(N,N)-G*B(N,N))/C
458 52 51 DO17 I=2,N
459 53 52 E=0.0
460 54 53 F=0.0
461 55 54 M=N+1-I
462 56 55 DO18 J=2,I
463 57 56 L=N+J-I
464 58 57 E=E+Y(L)*A(M,L)-Z(L)*B(M,L)
465 59 58 18 F=F+Y(L)*B(M,L)+Z(L)*A(M,L)
466 60 59 D=A(M,M)*A(M,M)+B(M,M)*B(M,M)
467 61 60 G=Y(M)
468 62 61 Y(M)=((G-E)*A(M,M)+(Z(M)-F)*B(M,M))/D
469 63 62 17 Z(M)=((Z(M)-F)*A(M,M)-(G-E)*B(M,M))/D
470 64 63 E=0.0
471 65 64 DO51 I=1,N
472 66 65 F=Y(I)*Y(I)+Z(I)*Z(I)
473 67 66 IF (E.GE.F)
474 68 67 5 GO TO 51
475 69 68 52 E=F
476 70 69 L=I
477 71 70 51 CONTINUE
478 72 71 A1=Y(L)
479 73 72 A2=Z(L)
480 74 73 DO14 I=1,N
481 75 74 G=Y(I)
482 76 75 Y(I)=(G*A1+Z(I)*A2)/E
483 77 76 14 Z(I)=(Z(I)*A1-G*A2)/E
484 78 77 IF (JJJ.EQ.4)
485 79 78 5 GO TO 6666
486 80 79 DO15 I=2,N
487 81 80 IF (INTER(I).NE.I)
488 82 81 5 GO TO 30
489 83 82 16 G=Y(I)
490 84 83 Y(I)=Y(I-1)
491 85 84 Y(I-1)=G
492 86 85 G=Z(I)
493 87 86 Z(I)=Z(I-1)
494 88 87 Z(I-1)=G
495 89 88 30 Y(I)=Y(I)+A(I,I-1)*Y(I-1)-B(I,I-1)*Z(I-1)
496 90 89 15 Z(I)=Z(I)+B(I,I-1)*Y(I-1)+A(I,I-1)*Z(I-1)
497 91 90 8888 CONTINUE
498 0 0 C EIGENVECTOR OF HESSENBURG MATRIX OBTAINED
499 92 91 6666 M=N-1
500 93 92 DO21 I=2,M
501 94 93 J=N+1-I
502 95 94 L=J+1
503 96 95 DO20 K=L,N
504 97 96 Y(K)=Y(K)-Y(J)*A(K,J-1)+Z(J)*B(K,J-1)
505 98 97 20 Z(K)=Z(K)-Z(J)*A(K,J-1)+Y(J)*B(K,J-1)
506 99 98 LL=INT(N-I)
507 100 99 IF (J.EQ.LL)
508 101 100 3 GO TO 21
509 102 101 19 G=Y(J)
510 103 102 Y(J)=Y(LL)
511 104 103 Y(LL)=G
512 105 104 G=Z(J)
513 106 105 Z(J)=Z(LL)
514 107 106 Z(LL)=G

```



```

511 108 107 21 CONTINUE
512 0 0 C UNNORMALIZED EIGENVECTOR OBTAINED
513 109 108 E=0.0
514 110 109 DO22 I=1,N
515 111 110 F=Y(I)*Y(I)+Z(I)*Z(I)
516 112 111 IF (E.GE.F)
516 113 112 $ GO TO 22
517 114 113 23 E=F
518 115 114 L=I
519 116 115 22 CONTINUE
520 117 116 A1=Y(L)
521 118 117 A2=Z(L)
522 119 118 DO24 I=1,N
523 120 119 G=Y(I)
524 121 120 Y(I)=(G*A1+Z(I)*A2)/E
525 122 121 24 Z(I)=(Z(I)*A1-G*A2)/E
526 0 0 C NORMALIZED EIGENVECTOR OBTAINED
527 0 0 C-PUT Y(I) (REAL PART) IN EIGVECT(NUMOFEV,I)
528 123 122 DO 100 I=1,N
529 124 123 100 EIGVEC(NUM,I)=Y(I)
530 125 124 NUM=NUM+1
531 0 0 C PUT Z(I) (COMPLEX PART) IN EIGVECT(NUMOFEV,I)
532 126 125 DO 101 I=1,N
533 127 126 101 EIGVEC(NUM,I)=Z(I)
534 128 127 RETURN
535 129 1 END

```

UNIVERSITY OF CALIFORNIA LIBRARY

CALL GRAPH TABLE ENTRIES

SUBPROGRAM NAME= UNSEIG
 PROCESSED AS ITEM NUMBER 1 IN THE INPUT FILE
 NUMBER OF EXTERNAL CALLS= 5
 NAMES OF CALLED

IMVEC
 CALCEV
 COMPIT
 REALIT
 RVEC

NUMBER OF CALLERS= 0

SUBPROGRAM NAME= CALCEV
 PROCESSED AS ITEM NUMBER 2 IN THE INPUT FILE
 NUMBER OF EXTERNAL CALLS= 0
 NAMES OF CALLERS
 UNSEIG

SUBPROGRAM NAME= COMPIT
 PROCESSED AS ITEM NUMBER 4 IN THE INPUT FILE
 NUMBER OF EXTERNAL CALLS= 0
 NAMES OF CALLERS
 UNSEIG

SUBPROGRAM NAME= REALIT
 PROCESSED AS ITEM NUMBER 3 IN THE INPUT FILE
 NUMBER OF EXTERNAL CALLS= 0
 NAMES OF CALLERS
 UNSEIG

SUBPROGRAM NAME= RVEC
 PROCESSED AS ITEM NUMBER 5 IN THE INPUT FILE
 NUMBER OF EXTERNAL CALLS= 0
 NAMES OF CALLERS
 UNSEIG

SUBPROGRAM NAME= IMVEC
 PROCESSED AS ITEM NUMBER 6 IN THE INPUT FILE
 NUMBER OF EXTERNAL CALLS= 0
 NAMES OF CALLERS
 UNSEIG

NEXT LEAF IS FILE ENTRY NUMBER 2
 NEXT LEAF IS FILE ENTRY NUMBER 3
 NEXT LEAF IS FILE ENTRY NUMBER 4
 NEXT LEAF IS FILE ENTRY NUMBER 5
 NEXT LEAF IS FILE ENTRY NUMBER 6
 NEXT LEAF IS FILE ENTRY NUMBER 1

PHASE1 COMPLETE

PHASE2 COMPLETE

DIAGNOSTICS FOR PROGRAM UNIT CALCEV

***** NO ERRORS

DIAGNOSTICS FOR PROGRAM UNIT REALIT

***** NO ERRORS

DIAGNOSTICS FOR PROGRAM UNIT COMPIT

***** NO ERRORS

DIAGNOSTICS FOR PROGRAM UNIT RVEC

***** NO ERRORS

DIAGNOSTICS FOR PROGRAM UNIT IMVEC

***** NO ERRORS

DIAGNOSTICS FOR PROGRAM UNIT UNSEIG

***** NO ERRORS

DIAGNOSTICS FOR PROGRAM UNIT CALCEV

*****WARNING NO. 27

THE LOCAL VARIABLE NAMED C IS ASSIGNED A VALUE IN
BLOCK 11 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.
ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

11 12 13 14 15 34 35 36 37 38
39 40 41 42

ORIGINAL PAGE IS
OF POOR QUALITY

15X

DIAGNOSTICS FOR PROGRAM UNIT REALIT

***** NO WARNINGS

DIAGNOSTICS FOR PROGRAM UNIT COMPIT

***** NO WARNINGS

DIAGNOSTICS FOR PROGRAM UNIT RVEC

*****WARNING NO. 19

A VARIABLE IN A PARAMETER LIST IS USED FOR NEITHER INPUT NOR
OUTPUT.

NAME OF VARIABLE NVEC

DIAGNOSTICS FOR PROGRAM UNIT IMVEC

*****WARNING NO. 19

A VARIABLE IN A PARAMETER LIST IS USED FOR NEITHER INPUT NOR
OUTPUT.

NAME OF VARIABLE NVEC

*****WARNING NO. 27

THE LOCAL VARIABLE NAMED C IS ASSIGNED A VALUE IN
BLOCK 15 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.
ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

15 16 17 18 19 20 21 22 23 24
25 27 28 35 36 37

16K

*****WARNING NO. 27

THE LOCAL VARIABLE NAMED L IS ASSIGNED A VALUE IN
BLOCK 114 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

114 115 110 111 113 114

DIAGNOSTICS FOR PROGRAM UNIT UNSEIG

*****WARNING NO. 16 STATEMENT NO. 150 BASIC BLOCK NO. 149

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DIMENSIONALITY.

CALLING SUBPROGRAM CALLED SUBPROGRAM

UNSEIG

REALIT

ARGUMENT POSITION

6

6

NUMBER OF DIMENSIONS

0

1

NAME OF ARGUMENT

CC

C

KIND OF ARGUMENT

IDENTIFIER

INPUT CLASS

NON INPUT

OUTPUT CLASS

STR.OUTPUT

*****WARNING NO. 16 STATEMENT NO. 150 BASIC BLOCK NO. 149

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DIMENSIONALITY.

CALLING SUBPROGRAM CALLED SUBPROGRAM

UNSEIG

REALIT

ARGUMENT POSITION

7

7

NUMBER OF DIMENSIONS

0

1

NAME OF ARGUMENT

SS

S

KIND OF ARGUMENT

IDENTIFIER

INPUT CLASS

NON INPUT

OUTPUT CLASS

STR.OUTPUT

*****WARNING NO. 10 STATEMENT NO. 166 BASIC BLOCK NO. 165

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DATA TYPES.

CALLING SUBPROGRAM CALLED SUBPROGRAM

UNSEIG

RVEC

ARGUMENT POSITION

10

10

17K

DATA TYPE	REAL	INTEGER
NAME OF ARGUMENT	CC	INTER
KIND OF ARGUMENT	IDENTIFIER	
INPUT CLASS		NON INPUT
OUTPUT CLASS		STR.OUTPUT

****WARNING NO. 16 STATEMENT NO. 166 BASIC BLOCK NO. 165

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DIMENSIONALITY.

CALLING SUBPROGRAM CALLED SUBPROGRAM

	UNSEIG	RVEC
ARGUMENT POSITION	10	10
NUMBER OF DIMENSIONS	0	1
NAME OF ARGUMENT	CC	INTER
KIND OF ARGUMENT	IDENTIFIER	
INPUT CLASS		NON INPUT
OUTPUT CLASS		STR.OUTPUT

****WARNING NO. 16 STATEMENT NO. 166 BASIC BLOCK NO. 165

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DIMENSIONALITY.

CALLING SUBPROGRAM CALLED SUBPROGRAM

	UNSEIG	RVEC
ARGUMENT POSITION	11	11
NUMBER OF DIMENSIONS	0	1
NAME OF ARGUMENT	SS	Y
KIND OF ARGUMENT	IDENTIFIER	
INPUT CLASS		NON INPUT
OUTPUT CLASS		STR.OUTPUT

****WARNING NO. 10 STATEMENT NO. 171 BASIC BLOCK NO. 170

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DATA TYPES.

CALLING SUBPROGRAM CALLED SUBPROGRAM

	UNSEIG	IMVEC
ARGUMENT POSITION	11	11
DATA TYPE	REAL	INTEGER
NAME OF ARGUMENT	CC	INTER
KIND OF ARGUMENT	IDENTIFIER	
INPUT CLASS		NON INPUT
OUTPUT CLASS		STR.OUTPUT

****WARNING NO. 16 STATEMENT NO. 171 BASIC BLOCK NO. 170

18K

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DIMENSIONALITY.

CALLING SUBPROGRAM CALLED SUBPROGRAM

	UNSEIG	IMVEC
ARGUMENT POSITION	11	11
NUMBER OF DIMENSIONS	0	1
NAME OF ARGUMENT	CC	INTER
KIND OF ARGUMENT	IDENTIFIER	
INPUT CLASS		NON INPUT
OUTPUT CLASS		STR.OUTPUT

*****WARNING NO. 16 STATEMENT NO. 171 BASIC BLOCK NO. 170

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DIMENSIONALITY.

CALLING SUBPROGRAM CALLED SUBPROGRAM

	UNSEIG	IMVEC
ARGUMENT POSITION	12	12
NUMBER OF DIMENSIONS	0	1
NAME OF ARGUMENT	SS	Y
KIND OF ARGUMENT	IDENTIFIER	
INPUT CLASS		NON INPUT
OUTPUT CLASS		STR.OUTPUT

*****WARNING NO. 16 STATEMENT NO. 171 BASIC BLOCK NO. 170

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DIMENSIONALITY.

CALLING SUBPROGRAM CALLED SUBPROGRAM

	UNSEIG	IMVEC
ARGUMENT POSITION	13	13
NUMBER OF DIMENSIONS	0	1
NAME OF ARGUMENT	Z	Z
KIND OF ARGUMENT	IDENTIFIER	
INPUT CLASS		NON INPUT
OUTPUT CLASS		STR.OUTPUT

*****WARNING NO. 16 STATEMENT NO. 171 BASIC BLOCK NO. 170

CORRESPONDING ARGUMENTS IN THE PARAMETER LISTS ARE OF
DIFFERENT DIMENSIONALITY.

CALLING SUBPROGRAM CALLED SUBPROGRAM

	UNSEIG	IMVEC
ARGUMENT POSITION	14	14
NUMBER OF DIMENSIONS	0	2
NAME OF ARGUMENT	BB	R
KIND OF ARGUMENT	IDENTIFIER	
INPUT CLASS		NON INPUT
OUTPUT CLASS		STR.OUTPUT

ORIGINAL PAGE IS
OF POOR QUALITY

*****WARNING NO. 27

THE LOCAL VARIABLE NAMED GREAT IS ASSIGNED A VALUE IN
BLOCK 13 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.
ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

13	14	15	16	17	18	27	28	29	33
34	35	36	39	40	41	42	43	44	45
46	47	48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63	64	66
67	69	70	71	72	76	77	78	79	80
81	84	86	87	151	152	175			

*****WARNING NO. 27

THE LOCAL VARIABLE NAMED MAX IS ASSIGNED A VALUE IN
BLOCK 14 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.
ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

14	15	10	11	13	14
----	----	----	----	----	----

*****WARNING NO. 27

THE LOCAL VARIABLE NAMED C IS ASSIGNED A VALUE IN
BLOCK 108 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.
ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

108	109	110	111	112	120	129	132	133	135
136	138	139	142	143	144	147	148	149	150
91	92	93	94	95	97	98	102	103	107
108									

*****WARNING NO. 27

THE LOCAL VARIABLE NAMED DELTA IS ASSIGNED A VALUE IN
BLOCK 76 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.
ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

76	77	78	79	80	81	84	86	87	151
152	175								

*****WARNING NO. 27

THE LOCAL VARIABLE NAMED AR1 IS ASSIGNED A VALUE IN
BLOCK 148 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS
 148 149 150 91 92 93 94 95 97 98
 102 104 105 106 80 81 84 85 88

 *****WARNING NO. 27

THE LOCAL VARIABLE NAMED AR3 IS ASSIGNED A VALUE IN
 BLOCK 124 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
 USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.
 ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS
 124 125 126 127 91 92 93 94 95 97
 98 102 104 105 106 80 81 84 85 88
 89

 *****WARNING NO. 27

THE LOCAL VARIABLE NAMED AR4 IS ASSIGNED A VALUE IN
 BLOCK 125 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
 USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.
 ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS
 125 126 127 91 92 93 94 95 97 98
 102 104 105 106 80 81 84 85 88 89
 90

 *****WARNING NO. 27

THE LOCAL VARIABLE NAMED ANUM IS ASSIGNED A VALUE IN
 BLOCK 115 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
 USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON ALL PATHS.
 ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS
 115 116 117 118 122 123 124 125 126 127
 91 92 93 94 95 97 98 102 103 107
 108 109 110 111 113 114 115

 *****WARNING NO. 27

THE LOCAL VARIABLE NAMED RHO IS ASSIGNED A VALUE IN
 BLOCK 122 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
 USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.
 ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS
 122 123 124 125 126 127 91 92 93 94
 95 97 98 102 103 107 108 109 110 111
 113 114 115 116 117 118 122

 *****WARNING NO. 27

THE LOCAL VARIABLE NAMED SIGMA IS ASSIGNED A VALUE IN

21K

BLOCK 123 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

123 124 125 126 127 91 92 93 94 95
97 98 102 103 107 108 109 110 111 113
114 115 116 117 118 122 123

****WARNING NO. 27

THE LOCAL VARIABLE NAMED JI IS ASSIGNED A VALUE IN
BLOCK 171 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

171 172 173 175

****WARNING NO. 27

THE LOCAL VARIABLE NAMED NM IS ASSIGNED A VALUE IN
BLOCK 156 AND IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT
USE OR IS NOT SUBSEQUENTLY USED AS INPUT, ON ALL PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

156 157 158 159 160 161 162 163 164 170
171 172 173 175

****MESSAGE NO. 37

SUBPROGRAM UNSEIG IS NEVER CALLED.

DIAGNOSTICS FOR PROGRAM UNIT CALCEV

****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED VEC
IS ASSIGNED A VALUE IN BLOCK 39 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

39 40

CLASSIFICATION OF PARAMETER AND COMMON VARIABLES

SUBROUTINE CALCEV

PARAMETERS/ENTRIES

ORDER NAME INPUT CLASS OUTPUT CLASS

22K

		STRICT	OUTPUT
1	A	STRICT	NON
2	NMAX	STRICT	NON
3	DEL	STRICT	NON
4	VEC	NON	STRICT
5	IJK	STRICT	STRICT
6	N	STRICT	NON
7	M	STRICT	NON

DIAGNOSTICS FOR PROGRAM UNIT REALIT

 *****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED C
 IS ASSIGNED A VALUE IN BLOCK 11 AND SOME ELEMENT OF THE
 ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
 THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

11 12 13 23 8 9 10 14

 *****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED S
 IS ASSIGNED A VALUE IN BLOCK 12 AND SOME ELEMENT OF THE
 ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
 THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

12 13 23 8 9 10 14 15

CLASSIFICATION OF PARAMETER AND COMMON VARIABLES

SUBROUTINE REALIT

ORDER	NAME	PARAMETERS/ENTRIES INPUT CLASS	OUTPUT CLASS
1	A	STRICT	STRICT
2	NMAX	STRICT	NON
3	L	STRICT	NON
4	N	STRICT	NON
5	SHIFT	STRICT	NON
6	C	NON	STRICT
7	S	NON	STRICT

DIAGNOSTICS FOR PROGRAM UNIT COMPIT

 *****MESSAGE NO. 42

ORIGINAL PAGE IS
 OF POOR QUALITY

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED A
IS ASSIGNED A VALUE IN BLOCK 66 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

66 67

CLASSIFICATION OF PARAMETER AND COMMON VARIABLES

SUBROUTINE , COMPIT

ORDER	NAME	PARAMETERS/ENTRIES INPUT CLASS	OUTPUT CLASS
1	A	STRICT	OUTPUT
2	NMAX	STRICT	NON
3	L	STRICT	NON
4	N	STRICT	NON
5	RHO	INPUT	NON
6	SIGMA	INPUT	NON

DIAGNOSTICS FOR PROGRAM UNIT RVEC

*****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED A
IS ASSIGNED A VALUE IN BLOCK 11 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

11 12

*****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED EIGVEC
IS ASSIGNED A VALUE IN BLOCK 77 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

77 77

*****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED INTER
IS ASSIGNED A VALUE IN BLOCK 14 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

14 15 16 21 5 6 14

 *****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED Y
 IS ASSIGNED A VALUE IN BLOCK 65 AND SOME ELEMENT OF THE
 ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
 THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

65 66

CLASSIFICATION OF PARAMETER AND COMMON VARIABLES

SUBROUTINE RVEC

ORDER	NAME	PARAMETERS/ENTRIES INPUT CLASS	OUTPUT CLASS
1	A	STRICT	STRICT
2	NMAX	STRICT	NON
3	INT	STRICT	NON
4	CI	STRICT	NON
5	SMALL	STRICT	NON
6	N	STRICT	NON
7	NUM	STRICT	NON
8	EIGVEC	NON	STRICT
9	NVEC	NON	NON
10	INTER	NON	STRICT
11	Y	NON	STRICT

DIAGNOSTICS FOR PROGRAM UNIT IMVEC

 *****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED A
 IS ASSIGNED A VALUE IN BLOCK 20 AND SOME ELEMENT OF THE
 ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
 THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

20 21

 *****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED EIGVEC
 IS ASSIGNED A VALUE IN BLOCK 126 AND SOME ELEMENT OF THE
 ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
 THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

126 126

25K

****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED INTER
IS ASSIGNED A VALUE IN BLOCK 26 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS
26 27 28 35 9 10 11 12 26

****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED Y
IS ASSIGNED A VALUE IN BLOCK 102 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS
102 103

****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED Z
IS ASSIGNED A VALUE IN BLOCK 105 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS
105 106

****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED B
IS ASSIGNED A VALUE IN BLOCK 23 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS
23 24

CLASSIFICATION OF PARAMETER AND COMMON VARIABLES

	SUBROUTINE	IMVEC	
ORDER	NAME	PARAMETERS/ENTRIES	OUTPUT CLASS
1	A	STRICT	STRICT
2	NMAX	STRICT	NON
3	INT	STRICT	NON
4	C1	STRICT	NON
5	AMU	STRICT	NON
6	SMALL	STRICT	NON
7	N	STRICT	NON
8	NUM	STRICT	STRICT
9	EIGVEC	NON	STRICT

10	NVEC	NON	NON
11	INTER	NON	STRICT
12	Y	NON	STRICT
13	Z	NON	STRICT
14	B	NON	STRICT

DIAGNOSTICS FOR PROGRAM UNIT UNSEIG

*****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED A
IS ASSIGNED A VALUE IN BLOCK 162 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

162 162

*****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED INT
IS ASSIGNED A VALUE IN BLOCK 16 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

16	17	18	27	28	29	33	34	35	36
39	40	5	6	7	8	9	10	11	12
15	16								

*****MESSAGE NO. 42

AN ELEMENT OF THE COMMON OR PARAMETER ARRAY NAMED D
IS ASSIGNED A VALUE IN BLOCK 43 AND SOME ELEMENT OF THE
ARRAY IS ASSIGNED A VALUE THEREAFTER BEFORE ANY INPUT USE OF
THE ARRAY ON SOME PATHS.

ONE SUCH PATH, INDICATED BY BASIC BLOCK NUMBERS, IS

43 43

CLASSIFICATION OF PARAMETER AND COMMON VARIABLES

SUBROUTINE UNSEIG

ORDER	NAME	PARAMETERS/ENTRIES INPUT CLASS	OUTPUT CLASS
1	A	STRICT	STRICT
2	NMAX	STRICT	NON
3	NN	STRICT	NON
4	NVEC	STRICT	NON
5	VEC	NON	STRICT
6	EIGVEC	NON	OUTPUT

ORIGINAL PAGE IS
OF POOR QUALITY

7
8
9
10
11
12

INT
CC
SS
Z
D
BB

NON
NON
NON
NON
NON
NON

STRICT
OUTPUT
OUTPUT
OUTPUT
STRICT
OUTPUT

ORIGINAL PAGE IS
OF POOR QUALITY

TOOL	IMPLEMENTATIONS	SOURCE LANGUAGES ACCEPTED	STATIC / DYNAMIC	
ATDG NASA-HOUSTON HOUSTON, TEXAS	UNIVAC	FORTRAN	STATIC	
FORTUNE CAPEX PHOENIX, ARIZONA	IBM CDC	FORTRAN	STATIC	
NBS ANALYZER NBS WASHINGTON, D.C.	UNIVAC	FORTRAN	STATIC DYNAMIC	
PET MCDONNELL- DOUGLAS HUNTINGTON BEACH CALIFORNIA	IBM CDC HONEYWELL G.E. UNIVAC	FORTRAN	STATIC DYNAMIC	DE
PFORT BELL LAB MURRAY HILL NEW JERSEY	IBM CDC BURROUGHS HONEYWELL OTHERS	FORTRAN	STATIC	
PPE BOOLE AND BABBAGE SUNNYVALE CALIFORNIA	IBM	ALL	DYNAMIC	
STRUCTURING ENGINE (SFORTAN) CANE, FARBER, GORDON, INC. PASADENA CALIFORNIA	IBM	FORTRAN	STATIC	
DISSECT MCDONNELL- DOUGLAS HUNTINGTON BEACH CALIFORNIA	PDP10	FORTRAN	STATIC	

IC / DYNAMIC	OPERATING COST 1-LOW 5-HIGH	EASE OF USE 1-EASY 5-DIFF.	FEATURES	LIMITATIONS
STATIC	4	3	.DETERMINES VALUES OF PROGRAM VARIABLES TO PRO- DUCE OPTIMAL TEST CASES	.DIFFICULT TO INTERPRET OUT- PUT
STATIC	2	1	.EXACT STMT EXECUTION COUNTS .ESTIMATED EXEC TIME FOR EACH STATEMENT .NUMBER OF IF STATEMENT TRUE PATHS TAKEN .SUBROUTINE EXECUTION TIMINGS	.NO LONGER ACTIVELY SUPPORTED
STATIC DYNAMIC	1	2	.SUMMARY OF SOURCE STATEMENTS BY FORTRAN TYPE .EXECUTION COUNTS FOR PROGRAM SEGMENTS	
STATIC DYNAMIC	2 - 3 DEPENDING ON OPTIONS	1	.EXACT STMT EXECUTION COUNTS .IF STATEMENT TRUE AND FALSE BRANCH COUNTS .MINIMUM/MAXIMUM VALUES ATTAINED BY PROGRAM VARIABLES .FIRST AND LAST VALUES ATTAINED BY PROGRAM VARIABLES .RELATIVE SUBROUTINE EXEC TIM- INGS	
STATIC	1	2	.CHECKS SOURCE PROGRAM FOR ADHERENCE TO A PORTABLE SUB- SET OF ANS FORTRAN .SUBPROGRAM COMMUNICATION CHECKED THROUGH COMMON & ARGUMENT LISTS .SUBPROGRAM CROSS REFERENCE OF ID TYPE,USAGE AND ATRIBUTES	
DYNAMIC	1	2	.SAMPLES EXECUTING PROGRAM AT REGULAR INTERVALS TO OBTAIN PERFORMANCE STATISTICS .MEASURES I/O WAIT TIME .MEASURES TIME SPENT EXECUTING SVC'S	.PERFORMANCE STATISTICS ARE NOT REPORTED AT SOURCE STMT. LEVEL
STATIC	5	1	.PRODUCES A STRUCTURED FORTRAN PROGRAM AND LISTING FROM FORTRAN SOURCE .PREPROCESSOR AVAILABLE TO TRANSLATE STFUCTURED PROGRAM INTO FORTRAN FOR COMPILATION	.1000K LOAD MODULE .CAN PRODUCE EXCESSIVELY LARGE STRUC- TURED PROGRAMS
STATIC	4	3	.TESTS PROGRAMS INFORMALLY BY COMPUTING VALUES OF PROGRAM VARIABLES ALONG SELECTED PATHS .ALLOWS SYMBOLIC EXECUTION OF PROGRAMS	.SOME USER TIME REQUIRED TO SET UP TEST CASES

MIC	OPERATING COST 1-LOW 2-HIGH	EASE OF USE 1-EASY 5-DIFF.	FEATURES	LIMITATIONS
	2	2	.CHECKOUT AND DOCUMENTATION AID .MICROFILM FLOWLIST WITH ALL TRANSFERS INDICATED BY ARROWS AND ALL DO LOOPS INDICATED BY BRACKETS .COMPLETE CROSS-REFERENCED PROGRAM GLOSSARY	
	5	2	.CHECKS FOR A WIDE VARIETY OF SOURCE ERRORS .ISSUES WARNING MESSAGES FOR POSSIBLE ERRORS	.REDUNDANT OUTPUT MESSAGES

A.6 RECOMMENDATIONS

A knowledge of currently available automated verification tools should be part of the software designers repertoire of techniques and procedures for developing reliable software. These tools can impact the software life cycle at five points:

- (1) During program development to insure program specifications and standards are met.
- (2) In formal acceptance testing as a record of acceptance standards.
- (3) As a debugging aid
- (4) As an integral part of program documentation
- (5) As a performance measurement tool aimed at improving program efficiency.

The following guidelines are recommended when considering an automated verification tool for use at GSFC.

- (1) Select the tool early in the software development cycle and have it installed at GSFC. Unforeseen delays often occur when installing tools at the users site.
- (2) Care must be taken to select a tool that is easy for the applications programmer to use. Clear and concise documentation must be readily available for the users.
- (3) The use of an automated verification tool must supplement, and not replace, good desk checking procedures and well designed test plans.
- (4) In most cases, the application of an automated verification tool increases machine costs. Many available tools are very powerful and can be misused by applying them to classes of problems that can be solved by simpler means.

The following are recommendations for the use of the tools involved in this study by GSFC.

PFORT is representative of a class of automated verification tools known as standards auditors. This tool checks FORTRAN programs for machine transferability, and can be especially valuable at GSFC when verifying that vendor programs coming into GSFC adhere to a prescribed set of standards.

PPE (Problem Program Evaluator) is currently in use at GSFC. Its approach to execution monitoring by interrupting running programs at preset time intervals is unique. PPE is a well established proven tool of considerable use in performance measurement applications.

The SFORT structuring engine is considered impractical at this time for use at GSFC due to extreme system resource requirements (1000K load module and high execution time).

The three execution analyzers (NBS Analyzer, FORTUNE and PET) can be grouped together functionally. FORTUNE is no longer actively supported. PET (Program Evaluator and Tester), although more costly than the NBS Analyzer, has been operating on IBM 360/370 computers since 1972 and offers powerful capabilities. Where cost is a major factor, the NBS Analyzer (not available on IBM) might be considered. In all other cases PET is recommended.

ATDG (Automatic Test Data Generator) and the DISSECT Symbolic Evaluation System can best be thought of as representative tools from current research into the field of program testing and proving. More research and development is required to fully realize tools of this kind and apply them toward a formal demonstration of program correctness. The usefulness of ATDG and DISSECT to GSFC at this time is doubtful, due to programmer time required to invoke the tool and interpret the output to draw meaningful conclusions about program correctness. However, continued refinement and development of these techniques warrant additional support.

JOYCE and DAVE have proven to be valuable documentation and checkout aids. JOYCE, however, is not currently available on IBM and therefore could not be installed at GSFC at this time.